

Mladi za napredek Maribora 2017

34. srečanje

Inovacijski predlog

Pametna soba 2

Elektrotehnika, elektronika

Avtor: ZORAN URAN
Mentor: SUZANA URAN
Šola: II. GIMNAZIJA MARIBOR

2017, Maribor

Mladi za napredek Maribora 2017

34. srečanje

Inovacijski predlog

Pametna soba 2

Elektrotehnika, elektronika

2017, Maribor

Kazalo

Povzetek	4
Zahvala	5
Uvod	6
Metodologija dela	7
0 Uvodna beseda.....	7
1 Zbiranje podatkov.....	7
1.1 Delovanje Bluetooth naprav [1]	7
1.2 Bluetooth modula HC05 in HC06 [1]	8
1.3 Spoznavanje Raspberry Pi računalnika.....	10
1.4 Svetlobni senzor	12
1.5 Operacijski ojačevalnik	13
1.6 Temperaturni senzor	15
1.7 Pregled tržišča	15
1.8 KNX vodilo	17
2 Načrtovanje	18
2.1 Bluetooth komunikacijska mreža	18
2.2 Osnova vsake naprave sistema Pametne hiše – novo vezje mikrokrmilnika	20
2.3 Vezje svetlobnega senzorja	22
2.4 Vezje temperaturnega senzorja	22
2.5 Stare električne inštalacije [1]	24
2.6 Načrtovanje programa mikrokrmilnika naprav v Pametni sobi	26
2.7 Načrtovanje programa Raspberry Pi računalnika.....	27
3 Izdelava tiskanih vezij in ožičenje	30
3.1 Priprava Raspberry Pi	30
3.2 Ožičenje	31
4 Preizkušanje in odprava napak.....	32

Rezultati.....	33
Razprava/Interpretacija rezultatov	37
Zaključek.....	39
Družbena odgovornost.....	40
Viri:	41
Dotatni viri (prebrana literatura):	42
Viri slike in razpredelnice:	43
Priloge.....	45
Priloga A: Ascii tabela	45
Priloga B: Tabela uporabljenih ukazov v poveljniškem načinu Bluetooth enot	46
PRILOGA C	47
C1. Opis izdelave elektronskih tiskov	47
C1.1 Izdelava elektronskega vezja (jedkanje,vrtanje)	47
C1.2 Spajkanje	50
C2. Opis programiranja.....	51
C3. Program na mikrokrmilniku vsake naprave:	53
C4. Program na Raspberry Pi računalniku	59
D. Računališke topologije	66

Povzetek

V tem inovacijskem predlogu sem izboljšal že obstoječi sistem pametne sobe, ki temelji na brezžični Bluetooth komunikaciji [1], a ga je omejevalo počasno vzpostavljanje povezave, potrebne za krmiljenje naprav (luč, žaluzije), ter pomanjkanje senzorjev. Sistem je deloval, ni pa bil uporabniško prijazen.

Cilj inovacij je narediti sistem pametnih hiš modularnejši in cenovno ugoden, da bodo postale pametne hiše dostopnejše invalidom in starejšim ljudem, ki jih najbolj potrebujejo.

Cilj sedanjega inovacijskega predloga je nadgraditi obstoječi sistem s hitrejšo komunikacijo, s senzorji za zaznavanje svetlobe in temperature, uvesti Raspberry Pi kot osrednji računalnik in sistem narediti uporabniško prijaznejši.

Nadgradnjo obstoječega sistema sem izvedel po svojih najboljših finančnih in tehničnih sposobnostih in preizkusil njegove tehniške lastnosti in uporabniško prijaznost. Nadgradnja je bila uspešna. Sistem je postal hitrejši in bolj praktičen ter omogoča avtomatsko krmiljenje.

Zahvala

Rad se bi zahvalil mentorici (moji mami) za nasvete in diagnoze mojih napak ter bratu za pomoč pri usposabljanju okolja Geany na Raspbery Pi. Zahvalil se bi tudi razredniku za lektoriranje mojega pisnega izdelka. In nazadnje se bi zahvalil moji koordinatorki za njeno vsestransko pomoč.

Uvod

V svojem predhodnem inovacijskem projektu Pametna soba [1] sem zgradil sistem pametne hiše, ki prejema ukaze iz pametnega telefona in jih preko brezžičnega vodila (Bluetooth HC-05, HC-06) posreduje napravam v moji sobi, ki »nekaj naredijo«, denimo vklop luči.

V tej moji Pametni sobi sem krmilil dve napravi: luč in žaluzije. V tej izvedbi je preprosta centrala na osnovi mikrokrmilnika PIC skrbela za prejemanje podatkov in preprosto mrežo Bluetooth piconet (glej podpoglavje 1.1).

Centrala je bila sestavljena iz mikrokrilnika in dveh bluetooth modulov, gospodarja in sužnja (glej podpoglavje 1.1). Na centralo sem se povezal z telefonom (takrat Amarino aplikacijo [13]) in z njim pošiljal ukaze za krmiljenje sobe. Centralni mikrokrmilnik pošlje AT povelje (glej podpoglavje 1.2) Bluetooth gospodarju. Bluetooth gospodar se poveže z ustrezno napravo in ukaz. Na mikrokrmilniku naprave je Bluetooth enota, ki deluje kot suženj, ki sprejme ukaze in nato ustrezno krmili napravo s pomočjo relejev preko digitalnih izhodov.

Vendar lahko Bluetooth piconet mreža zajema največ 7 naprav. Za nadaljnje večanje števila naprav v mreži pa bi se morala preoblikovati in povezati v mrežo scatternet (glej podpoglavje 1.1). Težava mreže Bluetooth piconet je bila tudi v tem, da je bila prepočasna, saj je ob pošiljanju ukaza preko mreže trajalo več sekund, preden se je vzpostavila zveza z drugo napravo v mreži in je bil ukaz posredovan. To je naredilo moj stari sistem v najboljšem primeru nepraktičen.

Moj cilj za ta inovacijski projekt je preprosta izboljšava sistema: odpraviti omejitve števila naprav, ki so lahko povezane v Bluetooth komunikacijsko mrežo, povečati hitrost prejemanja in razpošiljanja ukazov v tej mreži, še povečati modularnost sistema, vključiti v sistem pametne hiše senzorje (za svetlobo, temperaturo, ...), nadomestiti centralo na osnovi mikrokrmilnika PIC z računalnikom Raspberry Pi in omogočiti uporabniško prijazen sistem krmiljenja naprav.

Namen te inovacije je narediti sistem pametnih hiš modularnejši in cenejši, saj ne bo potrebno vgrajevanje žičnih inštalacij v hišo. To bo omogočilo navadnim ljudem in še posebej invalidom ter starejšim, ki ta sistem najbolj potrebujejo, da si ga vgradijo v svoj dom. Sistem pametne hiše pa bo mogoče naknadno tudi enostavno nadgrajevati.

Metodologija dela

0 Uvodna beseda

Ker je sedANJI inovacijski predlog nadaljevanje mojega predhodnega projekta, so metode in podatki zelo podobni. Takšni sklopi podatkov (npr. delovanje bluetooth komunikacije) bodo povzeti v mojem novem inovacijskem predlogu, vendar bo večja teža dana novim izvedbam in podatkom.

1 Zbiranje podatkov

Preden sem začel karkoli delati, sem moral zbrati podatke o napravah in sestavnih delih, ki jih bom uporabljal.

Tudi pri izvedbi Pametne sobe 2 sem uporabljal komunikacijo Bluetooth.

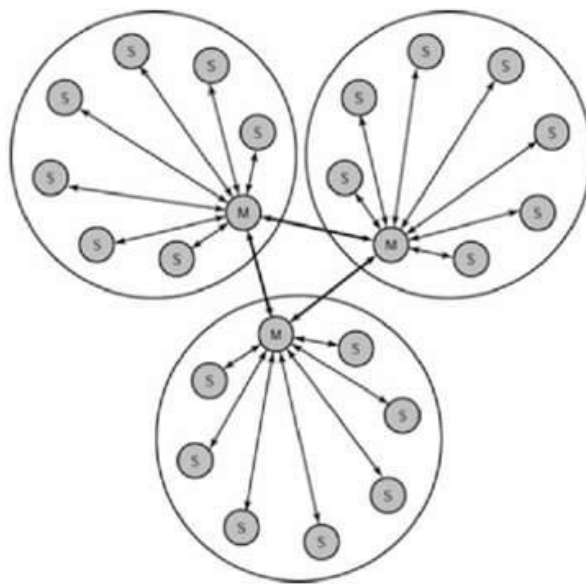
1.1 Delovanje Bluetooth naprav [1]

Komunikacija Bluetooth deluje na frekvencah 2.4 GHz ali 2.583GHz [11]. Bluetooth vsakih 625 ms naključno zamenja svoj 2MHz širok komunikacijski pas frekvenc in se s tem zagotavlja varnost prenosa in izogibanje motnjam iz okolice (preskakuje med 40 različnimi frekvencami). Zaradi relativno visoke frekvence komuniciranja je do met omejen na 1, 10 ali 100 metrov, odvisno od razreda enote bluetooth, ki določa moč oddajnika.

Pri komunikaciji Bluetooth se s frekvenco 2.4 GHz podatki prenašajo kot niz bitov (ki so bili prej pretvorjeni v nosilni val), ki se razdelijo v pakete (paket je dolg 625 ms) in pošljejo drugi enoti Bluetooth v mreži.

Lastnost Bluetooth komunikacijskega protokola je delitev naprav na dva vrsti: gospodarje (ang. master) in sužnje (ang. slave). Komunikacija je možna le med Bluetooth napravo, ki je gospodar, in Bluetooth napravo, ki je suženj. Naprava, ki je gospodar, lahko sproži komunikacijo in določa vzorec preskakovanja frekvenc med komunikacijo dveh Bluetooth enot. Bluetooth naprava suženj pa pošilja svoj identifikacijski naslov, sprejema zahteve za povezavo in prilagodi svoj vzorec preskakovanja frekvenc med komunikacijo vzorcu, ki ga določi naprava Bluetooth gospodar.

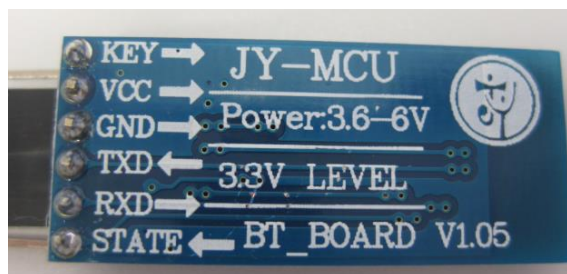
Bluetooth protokol omogoča povezovanje Bluetooth gospodarja z največ sedmimi Bluetooth napravami, ki so sužnji. Takšna mreža povezav se imenuje mreža piconet. Ta se lahko poveže v večjo mrežo scatternet, kot je prikazano na sliki 1. Na sliki 1 so s črko M označene Bluetooth enote, ki delujejo v načinu gospodar, s črko S pa Bluetooth enote, ki delujejo v načinu Bluetooth suženj.



Slika 1: Bluetooth scatternet mreža

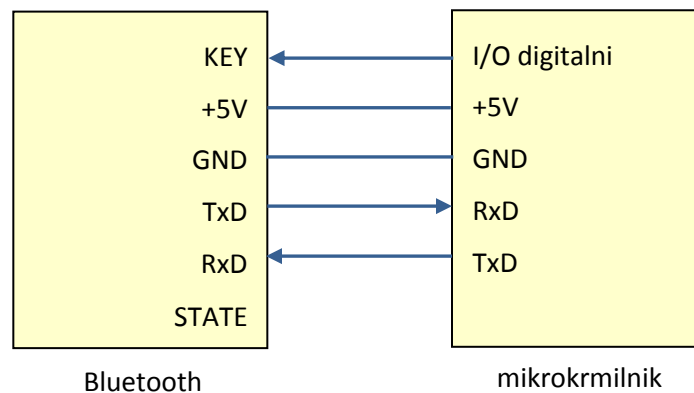
1.2 Bluetooth modula HC05 in HC06 [1]

V svojem sistemu pametne hiše uporabljam dokaj preproste enote Bluetooth serije HC, ki lahko nadomeščajo serijsko komunikacijo med napravami [12]. To serijo enot sem izbral zaradi ugodne cene in dobrih predhodnih izkušenj z njo. Uporabljal sem dve različni enoti iz te serije: enoto HC-06, ki je izključno Bluetooth suženj, in HC-05, ki lahko deluje kot Bluetooth gopodar ali Bluetooth suženj.



Slika 2: Bluetooth modul HC-05 (spredaj in zadaj)

Obe enoti izvajata Bluetooth (brežžično) komunikacijo samodejno, z mikrokrmilnikom pa komunicirata s pomočjo serijske komunikacije (serijskega prenosa podatkov). Zato Bluetooth enoti, ki pošilja podatke, s pomočjo serijske komunikacije posredujemo podatke, ki jih mora poslati, po uspešni povezavi, drugi Bluetooth enoti. Podobno Bluetooth enota, ki sprejema podatke, le-te po sprejemu podatkov od druge Bluetooth enote posreduje mikrokrmilniku s pomočjo serijske komunikacije. Hitrost serijske komunikacije (hitrost prenašanja podatkov) obeh Bluetooth enot znaša po tovarniških nastavitvah 9600 bit/s (Baudov).



Slika 3: Pravilno povezava Bluetooth enote HC-05 na mikrokrmilnik

Na sliki 3 je prikazana vezava Bluetooth enote HC-05 na mikrokrmilnik. Za razliko od HC-05 ima Bluetooth enota HC-06 le 4 priključne nogice (+5V, GND, TxD, RxD), ki jih povežemo enako kot pri HC-05.

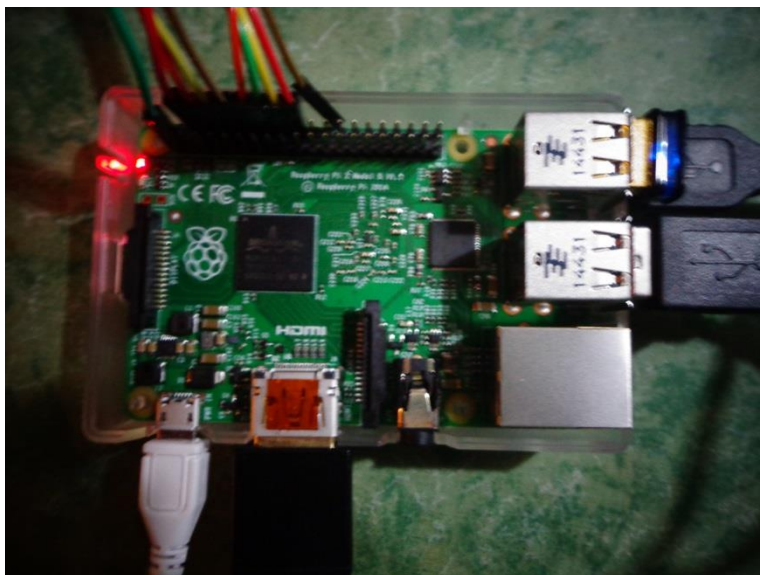
Omenil sem že, da lahko Bluetooth enota HC-05 deluje, kot Bluetooth gospodar ali Bluetooth suženj. Na Bluetooth enoti HC-05 obstaja nogica, ki se imenuje ključ (key)(Slika 2) in omogoča preklap enote v tako imenovan AT poveljniški način. Nakar lahko z AT (+nekaj) ukazi krmilimo delovanje Bluetooth enote. Z AT ukazom AT+ROLE=0 spremenimo delovanje enote iz načina Bluetooth gospodar v način Bluetooth suženj. Z AT ukazom AT+ROLE=1 pa spremenimo delovanje enote iz načina Bluetooth suženj v način Bluetooth gospodar.

Vzpostavitev Bluetooth komunikacije med dvema Bluetooth enotama je tudi relativno preprosta. S pomočjo ukaza AT+BIND+identifikacijska številka Bluetooth enote določimo s katero enoto, ki deluje v načinu Bluetooth suženj, naj se enota Bluetooth gospodar poveže. Povezava med obema Bluetooth enotama se nato izvede samodejno takoj, ko sta obe enoti pripravljeni.

Edina slabost Bluetooth enot serije HC je, da Bluetooth protokol ni popolnoma izkoriščen, saj lahko komunicira le z eno drugo napravo ob določenem času. Sicer je lahko »seznanjen« (pair-an) z sedmimi napravami (kar se še vedno imenuje piconet mreža).

1.3 Spoznavanje Raspberry Pi računalnika

V uvodu sem napisal, da si želim odpreti vrata do uporabniško prijaznih vmesnikov v mojem sistemu pametne hiše. Zato sem odločil za spremembo osrednjega računalnika v mojem sistemu pametne hiše in sicer sem zamenjal mikrokrmilnik PIC z večjim in boljšim računalnikom, ki omogoča gladko delovanje, lep prikaz podatkov, ima več priključkov za serijsko komunikacijo in podpira spletni strežnik (moji načrti za daljno prihodnost). Raspberry Pi ustreza vsem tem zahtevam, in še več.

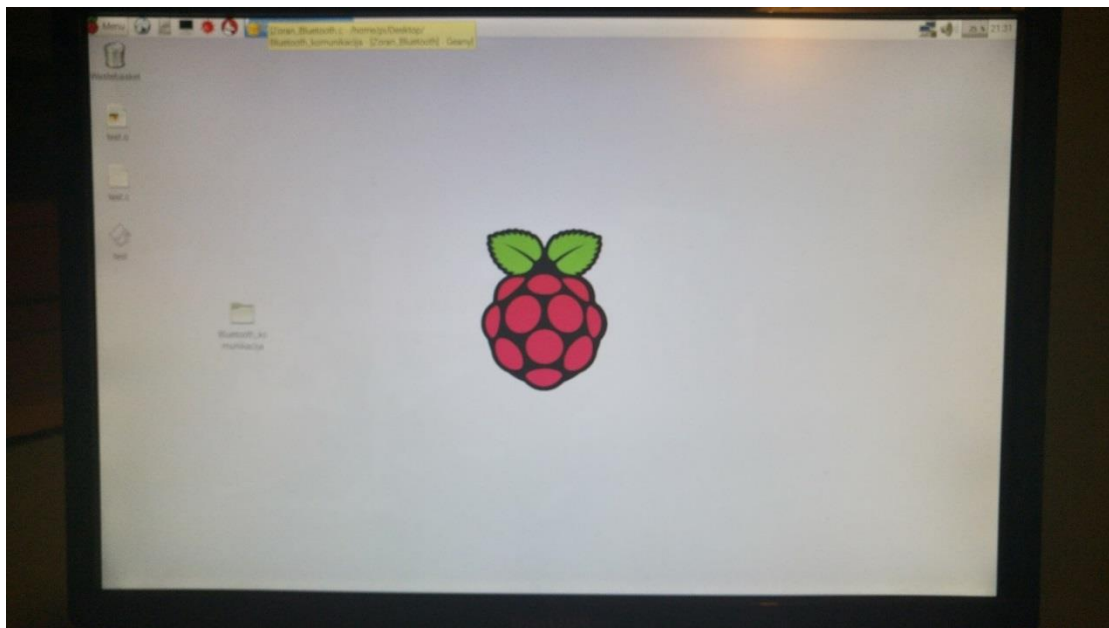


Slika 4: Raspberry Pi 2 B+ računalnik

Zanimanje za Raspberry Pi računalnik sem dobil, ko sem leta 2015 udeležil delavnice o uporabi Raspberry Pi računalnika na Srednji elektro-računalniški šoli (SERŠ-u) v Mariboru. Takrat sem ugotovil, da je računalnik Raspberry Pi zelo praktičen in ustreza mojim potrebam. Raspberry Pi računalnik je majhen (vgradili bi ga lahko skoraj kamorkoli) in poceni (cena od 40-60 EUR), podpira digitalne vhode in izhode ter serijsko komunikacijo, zanj je na voljo obilica odprto kodne programske opreme, kar mi omogoča, da lahko uporabim obstoječo programsko opremo in jo poljubno prilagodim svojim potrebam (samo strojno opremo računalnika težko spreminjamo zaradi visoke stopnje integracije).

Kaj sploh je Raspberry Pi? Raspberry Pi je majhen odprto kodni računalnik namenjen izobraževanju.[9] Za izobraževalni računalnik je zelo zmogljiv, saj ima en 900MHz ARM Cortex A-7 procesor s štiri jedri in 1GB spomina. Ima štiri priključke za USB, vrata s 40 digitalnimi vhodi/izhodi, ena serijska vrata, HDMI priključek, internetni priključek, 3.5 mm vtič za zvok, podpora mikro SD kartice,

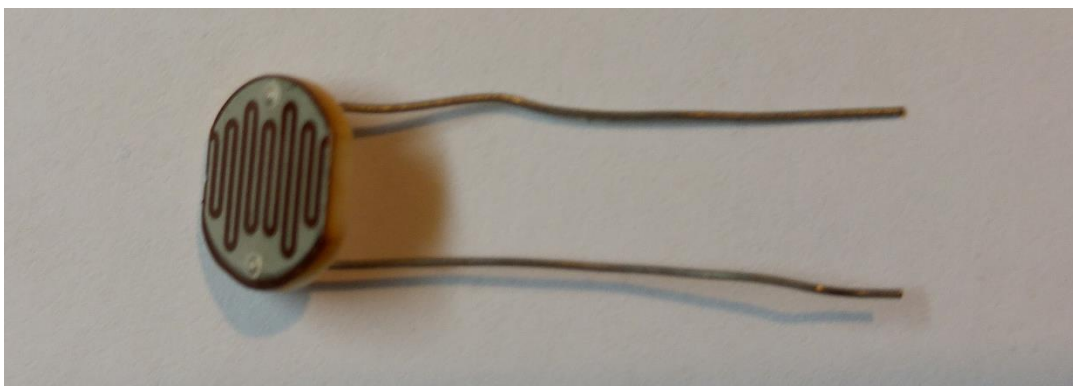
Raspberry Pi deluje na osnovi različice Linux operacijskega sistema, ki se imenuje Raspbian. Sicer podpira tudi druge operacijske sisteme, vendar je Raspbian specifično prilagojen za uporabo na Raspberry Pi računalniku. Skupaj z Raspbian operacijskim sistemom naložimo na svoj računalnik tudi mnoga programska okolja (python, scratch,...) in delujočo ukazno vrstico (LX terminal). Raspberry Pi okolje omogoča tudi preprosto dodajanje programskih okolij z ukazom apt-get install (npr. jaz sem si naložil programsko okolje za C++). Hkrati pa je (skoraj) tako preprost za uporabo kot nam običajni operacijski sistem Windows.



Slika 5: Namizje Raspberry Pi računalnika

1.4 Svetlobni senzor

Za merjenje svetlobe sem imel na izbiro fototranzistor in foto upor. Zaradi želje po natančnem branju svetlosti prostora (občutljivosti za svetlobo podobno človeškemu očesu) sem izbral fotoupor.



Slika 6: Primer fotoupora

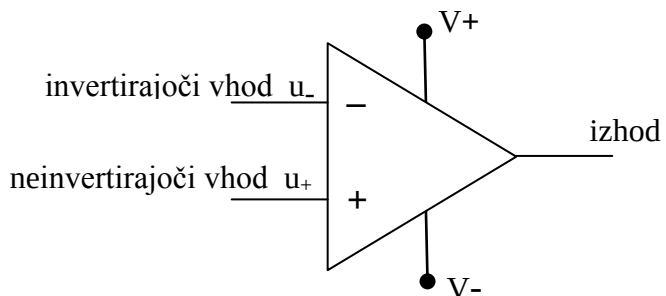
Fotoupor je elektronski element, ki spreminja upornost glede na osvetljenost. [4] Fotoupori so lahko narejeni iz polprevodnikov z visoko upornostjo. Za fotoupure narejene iz kadmijevega sulfida (CdS), je značilno, da so občutljivi na frekvence svetlobe v območju človeškega vida. Delovanje fotoupora temelji na dejstvu, da tanek plašč kadmijevega sulfida spreminja upornost zaradi energije, ki jo elektroni v kadmijevem sulfidu prejmejo od fotonov (delcev svetlobe). Elektroni, ki prejmejo energijo fotonov, lahko postanejo prosti elektroni in prevajajo elektriko. Zato se upornost kadmijevega sulfida pod vplivom svetlobe zmanjša.

Fotouporu, ki ga uporabljam, sem izmeril upornost v temi in ko je bil osvetljen. V temi je upornost fotoupora znašala $0.5\text{M}\Omega$, pri dnevni svetlobi $10\text{ k}\Omega$, pri močni osvetljenosti (v neposredni bližini žarnice) pa le 500Ω .

1.5 Operacijski ojačevalnik

Pozneje, v fazi načrtovanja sem ugotovil, da bom potreboval operacijski ojačevalnik za uspešno branje (analogno) spreminjajoče napetosti na fotouporu s pomočjo analognih vhodov mikrokontrolerja PIC. Za to nalogo sem izbral operacijski ojačevalnik LM358.

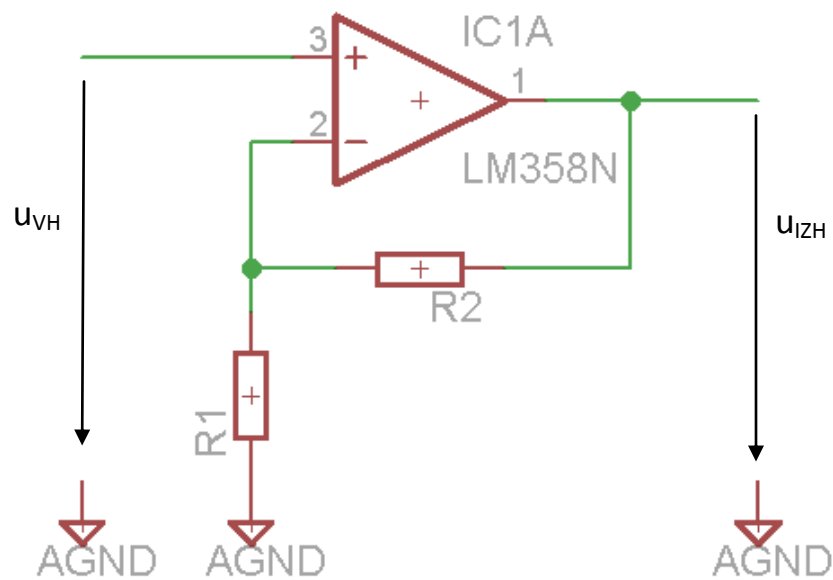
Operacijski ojačevalnik ima 5 priključkov. V+ in V- sta namenjena napajanju, vhod - je invertrani vhodu, vhod + pa neinvertirani vhod. Out je izhodna napetost (glej sliko 7).



Slika 7: Znak za operacijski ojačevalnik

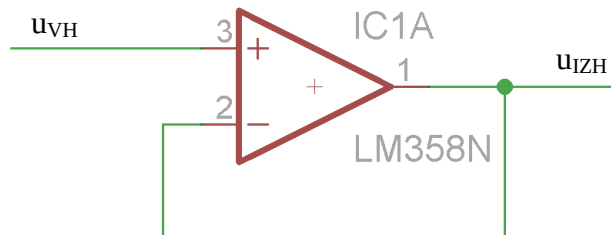
Operacijski ojačevalnik linearno ojačuje razliko napetosti med neinvertiranim in invertiranim vhodom (npr. 2V napetosti ojača na 4V). Stopnjo ojačanja K enostavno določimo z upori, ki so priključeni na drugi vhod ojačevalnika (slika 8). Vrednost ojačanja K je določena z enačbo 1.

$$K = 1 + \frac{R_2}{R_1} \quad (1)$$



Slika 8: Vezje neinvertirajočega ojačevalnika

Če v vezju neinvertirajočega ojačevalnika, prikazanega na sliki 8, izberemo vrednosti uporov $R1 = \infty$ in $R2 = 0$, tedaj ima ojačevalnik ojačanje 1 in vezje imenujemo sledilnik napetosti (slika 9). Dobra lastnost takšnega vezja je velika vhodna upornost.



Slika 9: Sledilnik napetosti

1.6 Temperaturni senzor

Za temperaturni senzor sem uporabil senzor DS18B20, ki deluje na osnovi polprevodnega silicija in lahko meri temperature od +125°C do -55°C. Senzor že ima vgrajen ojačevalnik in pretvornik za pretvorbo analognih signalov v digitalne. Zato je potrebno podatke temperaturnega sensorja le prebrati s posebno obliko komunikacije serijske komunikacije, ki uporablja »1-wire« protokol. Kot lahko sklepamo iz imena deluje »1-wire« protokol na samo eni žici. Tudi ta protokol pozna naprave, ki delujejo, kot gospodar, ali kot sužnj. Na eno žico lahko je lahko priključena le ena naprava gospodar, medtem, ko je lahko na to žico priklopljenih več naprav, ki delujejo, kot sužnji. Komunikacijo sproži naprava gospodar s signalom reset, naprave sužnji pa odgovorijo s signalom prisotnosti (oba signala sta opredeljena s kratkim pulzom). Za tem lahko naprava gospodar pošilja 8 bitne ukaze napravam sužnjem(v mojem primeru samo eni) in od njih prejme podatke.

Senzorju DS18B20 moramo poslati določeno zaporedje ukazov s katerim sprožimo meritev temperature, pretvorbo izmerjene vrednosti v digitalno, zapis v notranji pomnilnik, pošiljanje podatka in ponovno meritev.

Vrednost izmerjene temperature senzor posreduje mikrokontrolniku v obliki 9 bitnega ali 12 bitnega podatka, odvisno od nastavitvev.

1.7 Pregled tržišča

Pomemben komentar k mojemu prejšnjemu inovacijskem predlogu je bil pomanjkanje primerjave s podobnimi rešitvami na tržišču. Zato sem na tem mestu zbral v razpredelnici 1 podatke o sistemih za pametne hiše, ki jih lahko krmilimo s pomočjo pametnega telefona in so v tem oziru vsaj malo podobni mojemu (novemu in staremu sistemu).

Kot vidimo iz Razpredelnice 1 večina ponudnikov (Kopp, Gira, EQ-3), ki imajo brezžično povezane naprave uporabljaja radijsko izvedbo brezžične komunikacije, ki deluje na frekvencah 433MHz ali 868MHz. Predvideno je, da se brezžične naprave z ustreznim vmesnikom priključijo na KNX vodilo, ki je ogrodje sistema pametne hiše. Le podjetje eQ-3 ponuja radiatorske termostatske ventile, ki jih lahko krmilimo s pomočjo Bluetooth komunikacije. Vsi brezžični sistemi so napajani baterijsko kar lahko postane pri velikih objektih nepraktično.

Ponudba brezžičnih senzorjev za pametne hiše je velika. Ponudba brezžičnih naprav je razširjena z varnostnimi sistemi in krmiljenjem ogrevanja.

Razpredelnica 1:

Ime	Pametna soba 2	Pametna soba [1]	Seltron [6]	Kopp [7]	Gira [5]	EQ-3 [8]
Komunikacijska tehnologija	Bluetooth	Bluetooth	žična	Žična ali radijska	Žična ali radijska	Radijska ali bluetooth
Frekvenca/hitrost prenosa	2.4GHz	2.4GHz	žična	868MHz	433MHz	2.4GHz ali 868MHz
Doseg (brez ovir)	10m od naprave	10m od centrale	/	150m od oddajnika	150m od oddajnika	10m/100m od centrale
max število naprav	neomejeno	7	1	2-4 na sprejemnik	2-4 na sprejemnik	8 na centralo
Vrste naprav	Žaluzije, luč	Žaluzije, luč	Centralno ogrevanje	Žaluzije, luč, splošni relejski izhod, vtičnice	Žaluzije, luč, centralno ogrevanje	Radiatorski ventil
max število senzorjev	neomejeno	/	4	Veliko ?	Veliko ?	1
Vrste senzorjev	Temperaturni, svetlobni	/	Temperaturni, vlažnostni, svetlobni, zračnega tlaka	Varnostni, temperaturni, svetlobni	Temperaturni, svetlobni	temperaturni
Varnost	Preskakovanje frekvenc	Preskakovanje frekvenc	Žična komunikacija	Ena frekvenca	Preskakovanje frekvenc	Preskakovanje frekvenc/ena frekvenca
Dostop do internetnega oblaka	da	ne	da	da	da	Da (bluetoothne)
Način pošiljanja podatkov	Bluetooth zanka	Piconet mreža	žična	Senzor-centrala-naprava	Senzor-centrala-naprava	Piconet mreža/Centrala-naprava
Napajanje	Omrežno	Omrežno	Omrežno	Baterijsko	Baterijsko	Baterijsko
KNX standard	ne	ne	ne	da	da	ne

1.8 KNX vodilo

Z spoznavanjem tržišča sem ugotovil, da se 80% [2] vseh pametnih inštalacij uporablja KNX protokol za kumuniciranje med napravami.

KNX vodilo je žično vodilo na katerega so povezane vse naprave hkrati. Podatki se po tem vodilo pošiljajo po KNX protokolu. Naprave katerim so ti podatki namenjeni preberejo podatke ostale naprave jih prezrejo. KNX protokol podpira dvožično, trožično, radijsko in internetno povezavo.

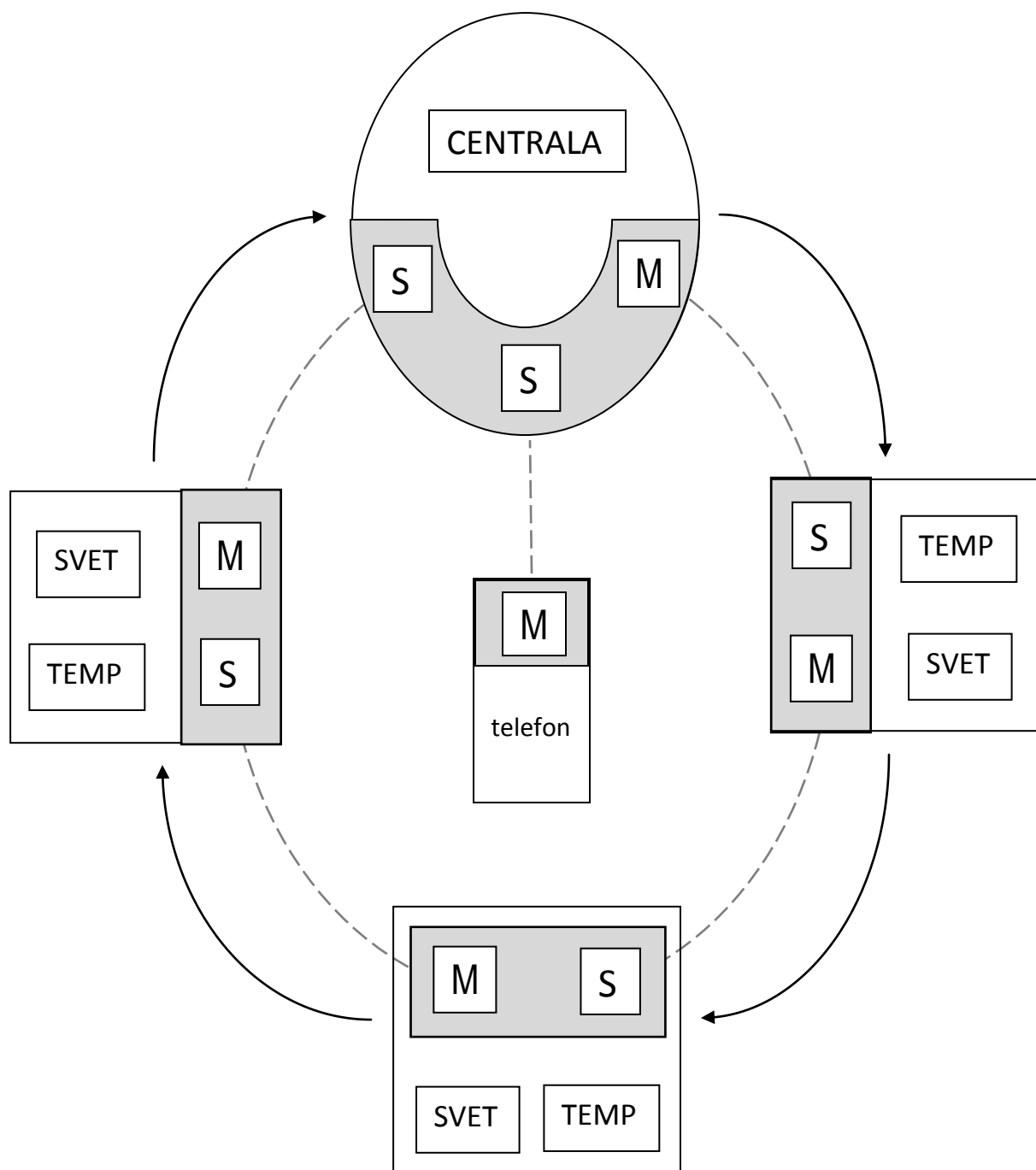
2 Načrtovanje

Načrtovanje Bluetooth brezžičnega komunikacijskega omrežja in senzorjev temperature ter svetlobe za izboljšan sistem pametne hiše.

2.1 Bluetooth komunikacijska mreža

Stara Bluetooth piconet mreža se je izkazala za prepočasno (še posebej v izvedbi z mojimi nizkocenovnimi Bluetooth moduli), zato sem se odločil, da jo bom izboljšal. Moje nezadovoljstvo z Bluetooth piconet mrežo izhaja iz dejstva, da je potrebno povezovanje, med enoto gospodar in enoto suženj, ki je potrebno za izvedbo komunikacije, traja precej dolgo - dobro sekundo. Prenos podatkov o merjenih veličinah v pametni sobi od vseh sedmih naprav sužnjev v Bluetooth piconet mreži do Bluetooth gospodarja, ki je osrednji mikroračunalnik, bi tako trajal kar nekaj sekund, saj se mora za vsak prenos podatka izvesti novo povezovanje Bluetooth enot.

Da bi odpravil navedeno počasnost Bluetooth komunikacijske mreže morajo biti vse Bluetooth enote v brezžični Bluetooth komunikacijski mreži ves čas povezane. To lahko dosežem s tem, da ima vsaka naprava sistema pametne hiše svoj mikrokrmilnik in dve Bluetooth enoti, eno Bluetooth enoto, ki deluje kot gospodar, in drugo Bluetooth enoto, ki deluje kot suženj (glej sliko 3). Tako zgrajene naprave sistema pametne hiše se lahko ves čas brezžično povežane, saj se v njej nahaja enako število enot Bluetooth gospodar in Bluetooth suženj. Tako nastala Bluetooth komunikacijska mreža ima obliko obroča (slika 10). V tej Bluetooth komunikacijski mreži prenašam vse podatke vedno le v eni smeri (izbral sem smer desno), da ne pride do izgube podatkov. Na sliki 10 pomeni M – enoto Bluetooth gospodar, S – enoto Bluetooth suženj, TEMP – senzor temperature in SVET – senzor svetlobe. Črtkane črte označujejo Bluetooth komunikacijo, puščice pa smer prenosa podatkov.



Slika 10: Bluetooth komunikacijska mreža v obliki obroča

Podatki, ki jih prenašam v tej Bluetooth mreži so v ASCII formatu kar pomeni, da ima vsaka črka ali znak pripisano svojo število (npr. velika črka A ima vrednost 65 – glej tabelo v prilogi). Podatki, ki jih pošiljam, so sestavljeni iz zaporedja črk ali znakov (npr. ABCTMP?).

Vsak podatek pa je sestavljen iz dveh delov predpona (ABC) in ukaza (TMP?). Prvi del podatka - predpona pove kateri napravi v mreži je ta ukaz namenjen ali od katere naprave prihaja (npr. ABC- je prva v nizu naprav). Drugi del podatka je ukaz, ki krmili dejanje naprave (npr.: TMP? – izmeri temperaturo in jo pošlji).

Predpostavimo, da je osrednji računalnik poslal po Bluetooth komunikacijski mreži ukaz: ABCTMP? V tem primeru naprava z imenom ABC, prepozna, da je ukaz namenjen njej, s pomočjo senzorja izmeri temperaturo in pošlje po komunikacijski mreži podatek, npr: ABC000401. Prvi del podatka - predpona pove od katere naprave prihaja podatek (npr. ABC- je prva v nizu naprav). Drugi del podatka je številka, ki predstavlja vrednost izmerjene temperature. To številko – vrednost izmerjene temperature je še potrebno pred izpisom na zaslon pretvoriti v °C.

Da lahko Bluetooth komunikacijska zanka pravilno deluje mora vsaka naprava poslati neprepoznane podatke, tiste ki niso zanjo, naprej naslednji napravi v obroču. Le te naprave lahko ta podatek prepoznajo. Tako se tudi izognem napačnemu delovanju naprav in zatikanju zaradi motenj, saj se pomankljiv (moten) ukaz pošilja naprej, do osrednje naprave, kjer se zbrši.

2.2 Osnova vsake naprave sistema Pametne hiše – novo vezje mikrokrmilnika

Za razliko od starih naprav moje Pametne sobe (razen osrednje naprave) morajo imeti nova tiskana vezja mikrokrmilnika dve Bluetooth enoti.

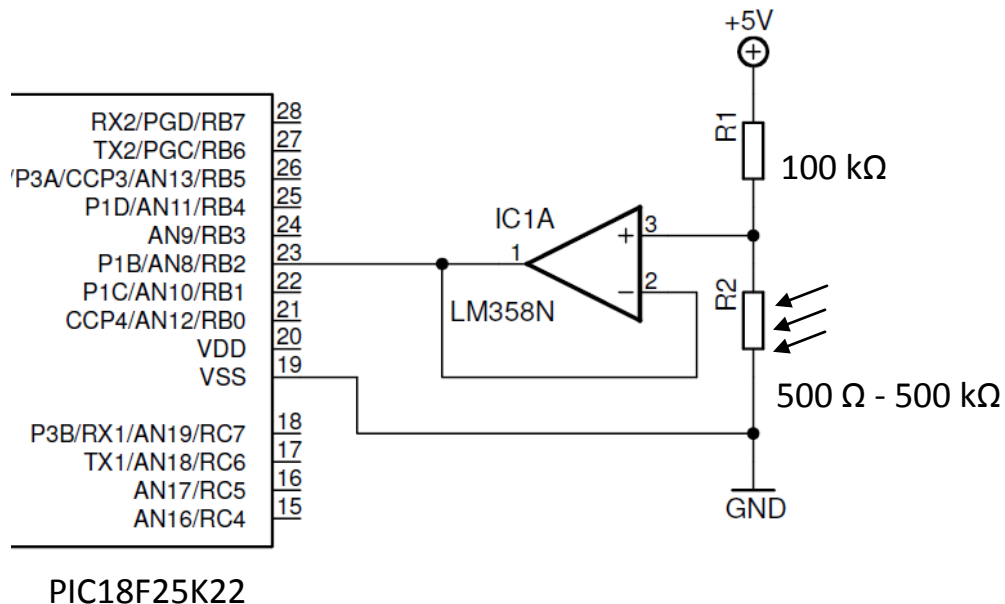
Pri risanju tiska sem upošteval naslednje zahteve za napravo sistema pametne hiše:

- vsi elementi morajo imeti 5V napajanje in biti prav priključeni,
- mikrokrmilnik mora imeti uro 8 MHz,
- mikrokrmilnik mora imeti reset tipko,
- vezje mora imeti priključek za programator
- mikrokrmilnik mora biti povezan z eno enoto Bluetooth suženj in z eno enoto Bluetooth gospodar, torej mora imeti dvojna UART serijska vrata,
- mikrokrmilnik mora imeti eno signalizacijsko svetilno (LED) diodo,
- vezje naj ima vsaj 4 digitalne/analogne vhode namenjene za senzorjema svetlobe in temperature,

2.3 Vezje svetlobnega senzorja

Osvetljenost v svoji sobi sem meril s pomočjo fotoupora. Fotoupor sem zvezal zaporedno z navadnim uporom in oba priključil na napetost. Tako sem dobil delilnik napetosti. Velikost dodatnega upora sem izbral tako, da se je napetost na fotouporu spreminjala od 0.5V, ko je bil fotoupor najmočnejše osvetljen, pa do 3.7V, ko je bil fotoupor v temi.

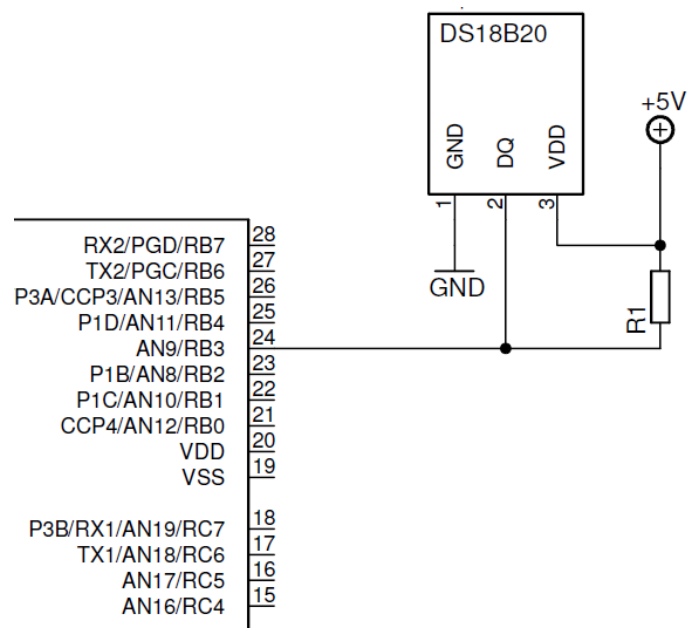
Na sliki 12 so predstavljene velikosti uporov in vezava delilnika napetosti na operacijski ojačevalnik, tako kot tudi vezava na analogni vhod mikrokrmilnika.



Slika 12: Priklop svetlobnega senzorja na mikrokrmilnik

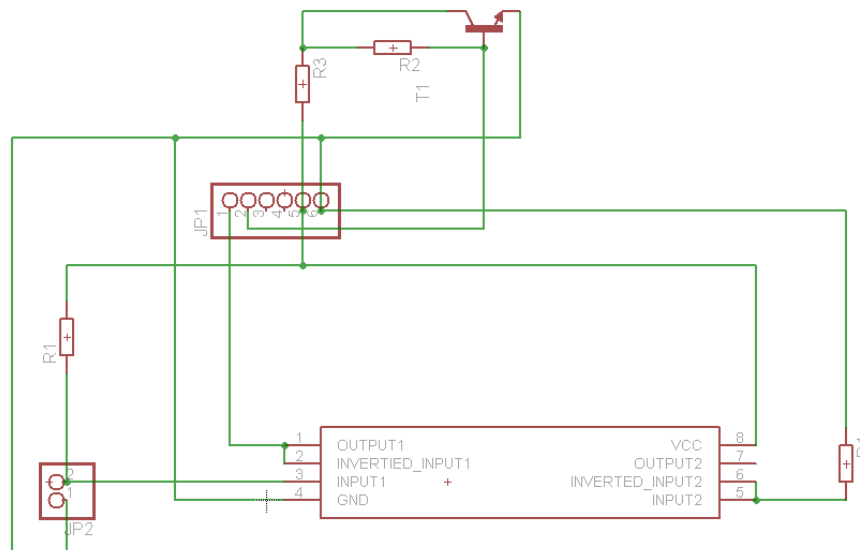
2.4 Vezje temperaturnega senzorja

Temperaturo sem pa meril s polprevodniškim termometrom DS18B20, ki že ima integrirano analogno digitalno pretvorbo. Po navodilih proizvajalca senzorja sem med komunikacijsko DQ nogico senzorja in +5V napajanjem vezal 4.7kΩ upor, tako kot prikazuje spodnje vezje. Nato pa sem DQ nogico senzorja še povezal na izbrani digitalni vhoda mikrokrmilnika.



Slika 13: Vezje temperaturnega senzorja

Tako kot vezje in tisk mikrokrmilnika sem narisal vezje in tisk svetlobnega in temperaturnega senzorja s programom Eagle (slika 14).

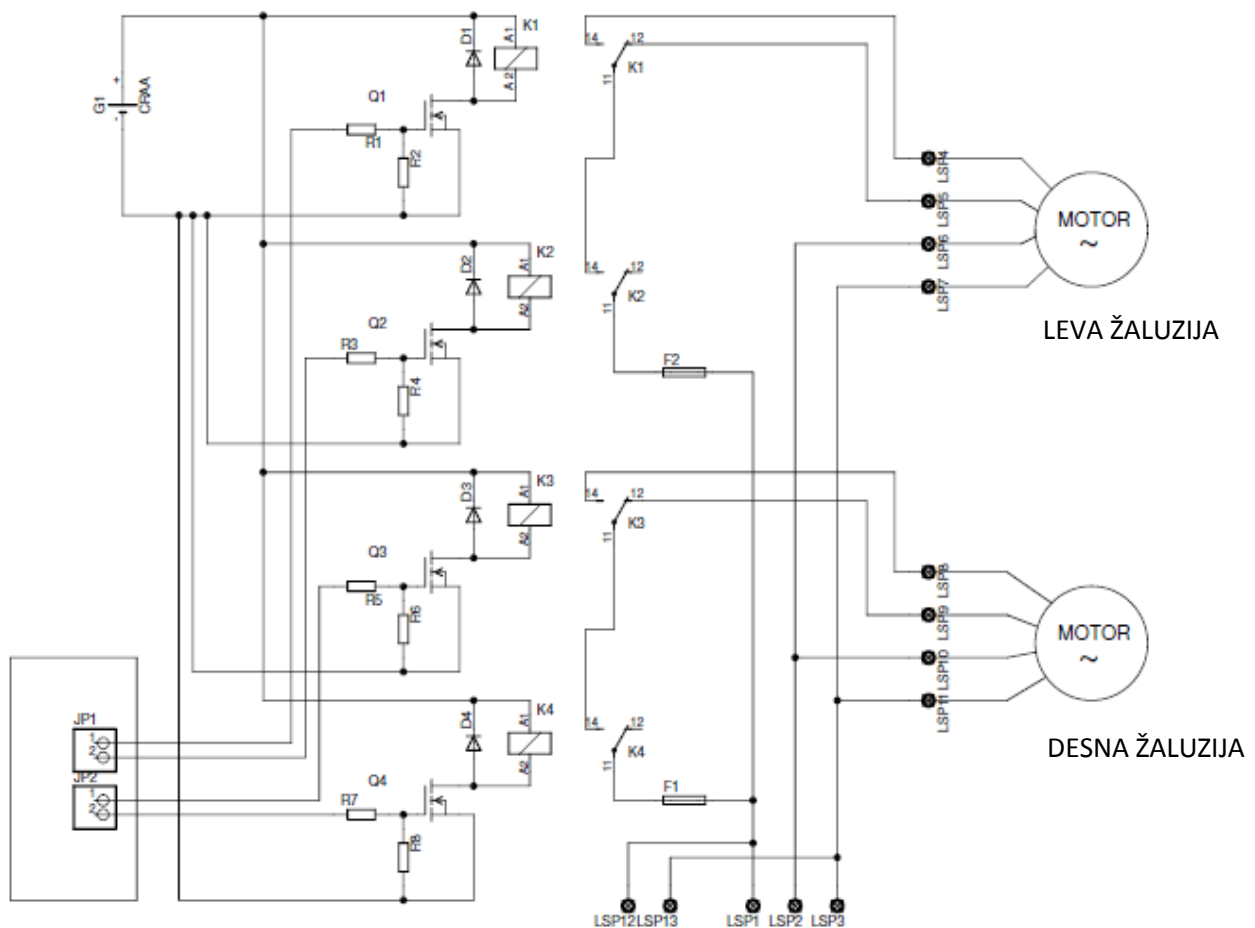


Slika 14: Vezje ploščice svetlobnega in temperaturnega senzorja

Opis vezja žaluzij:

V moji sobi se nahaja okno in balkonska vrata, zato ima moja soba dve žaluziji. Vsako žaluzijo krmili en motor. Krmilni sistem žaluzij je sestavljen iz 4 relejev in mikrokrmilnika. Tako kot pri luči je potrebno na vsakem digitalnem izhodu med mikrokrmilnikom in relejem vmesniško vezje FET tranzistorji.

Ploščica s tiskanim vezjem mikrokrmilnika, ki je pritrjena na zunanji strani škatle z releji (stari osrednji računalnik pametne sobe) pa krmili te releje. Na ploščico so tudi povezane tipke za ročno krmiljenje s katerimi lahko ročno dvigujem in spuščam žaluzije. Stanje dviganja in spuščanja žaluzij beleži mikrokrmilnik.



Slika 16: Vezje za krmiljenje motorjev žaluzij

Vse naprave, ki so povezane z Bluetooth mrežo, morajo imeti mikrokrmilnik z dvema Bluetooth enotama predstavljen v podpoglavju 2.2. Program za te mikrokrmilnike je napisan v programskem jeziku C. Osnovna struktura programa vseh naprav sistema Pametne hiše je enaka, čeprav se program spreminja od naprave do naprave. Vsaka naprava posreduje podatek naslednji Bluetooth napravi v komunikacijski mreži, če v prvem delu prejetega podatka ni prepoznala svojega imena. Razen svojega imena mora naprava v prejetem podatku prepoznati tudi naslednje ukaze:

-
- ```
graph TD; A[Prepoznavanje predpone] --> B[Prepoznavanje ukaza]; B --> C[Izvedba meritev]; C --> D[Pošiljanje novih podatkov]; D --> E[Prepošiljanje podatkov]; E --> A; E --> F[Branje podatkov]; F --> A; F --> B; F --> C; F --> D; F --> E; F --> F; F --> G[Prepoznavanje predpone];
```
- The diagram illustrates a data processing cycle in a control system. It consists of the following steps and connections:
- Prepoznavanje predpone** (Detection of the fault) leads to **Prepoznavanje ukaza** (Detection of the command).
  - Prepoznavanje ukaza** leads to **Izvedba meritev** (Execution of measurements).
  - Izvedba meritev** leads to **Pošiljanje novih podatkov** (Sending of new data).
  - Pošiljanje novih podatkov** leads to **Prepošiljanje podatkov** (Retransmission of data).
  - Prepošiljanje podatkov** leads back to **Prepoznavanje predpone**, completing the cycle.
  - Prepošiljanje podatkov** also leads to **Branje podatkov** (Reading of data).
  - Branje podatkov** leads to **Prepoznavanje predpone**, **Prepoznavanje ukaza**, **Izvedba meritev**, **Pošiljanje novih podatkov**, and **Prepošiljanje podatkov**.

26

V programu mikrokrmilnika, ki krmili luč, so bili k zgornjima dvema ukazoma dodani še trije dodatni ukazi za krmiljenje luči (koda programa za krmiljenje luči je bila vzeta iz starega programa in prirejena):

- LU1 (vklopi luč)
- LU0 (izklopi luč)
- LU? (Preberi stanje luči in jo pošlji podatek osrednjemu računalniku)

V programu mikrokrmilnika, ki krmili žaluzije, sta bila k zgornjima ukazoma za meritev temperature in osvetljenosti dodana dva dodatna ukaza za krmiljenje žaluzij (koda za krmiljenje žaluzij je bila izvzeta iz starega programa in prirejena):

- ŽALUAD000 (krmili desno žaluzijo)
- ŽALUAL000 (krmili levo žaluzijo)

Število na koncu ukaza pomeni koliko sekund vožnje naj bo žaluzija oddaljena od skrajne zgornje točke po izvedni tega ukaza.

Stara ploščica s tiskanim vezjem nima senzorjev, zato ne prepozna ukaza za branje senzorjev.

Koda vseh programov se nahaja v prilogi.

## **2.7 Načrtovanje programa Raspberry Pi računalnika**

Raspberry Pi računalnik lahko programiramo s pomočjo mnogih različnih programskih jezikov (python2, python3, scratch, ...). Jaz sem izbral programiranje v programskem jeziku C++ in razvojnem okolju z imenom Geany. Za programiranje sem uporabil knjižnico wiringpi [10], ki mi omogoča preprosto uporabo serijske komunikacije. Programiranje je bilo zelo podobno programiranju mojih mikrokrmilnikov.

V okviru tega inovacijskega predloga sem se pri pisanju programa za osrednji računalnik mojega sistema Pametne hiše (Raspberry Pi) omejil na vključitev mojega mobilnega telefona v Bluetooth komunikacijsko mrežo sistema Pametne hiše.

Osrednji računalnik mojega sistema Pametne hiše ima od vseh naprav najbolj zapleten program, saj mora le-ta program:

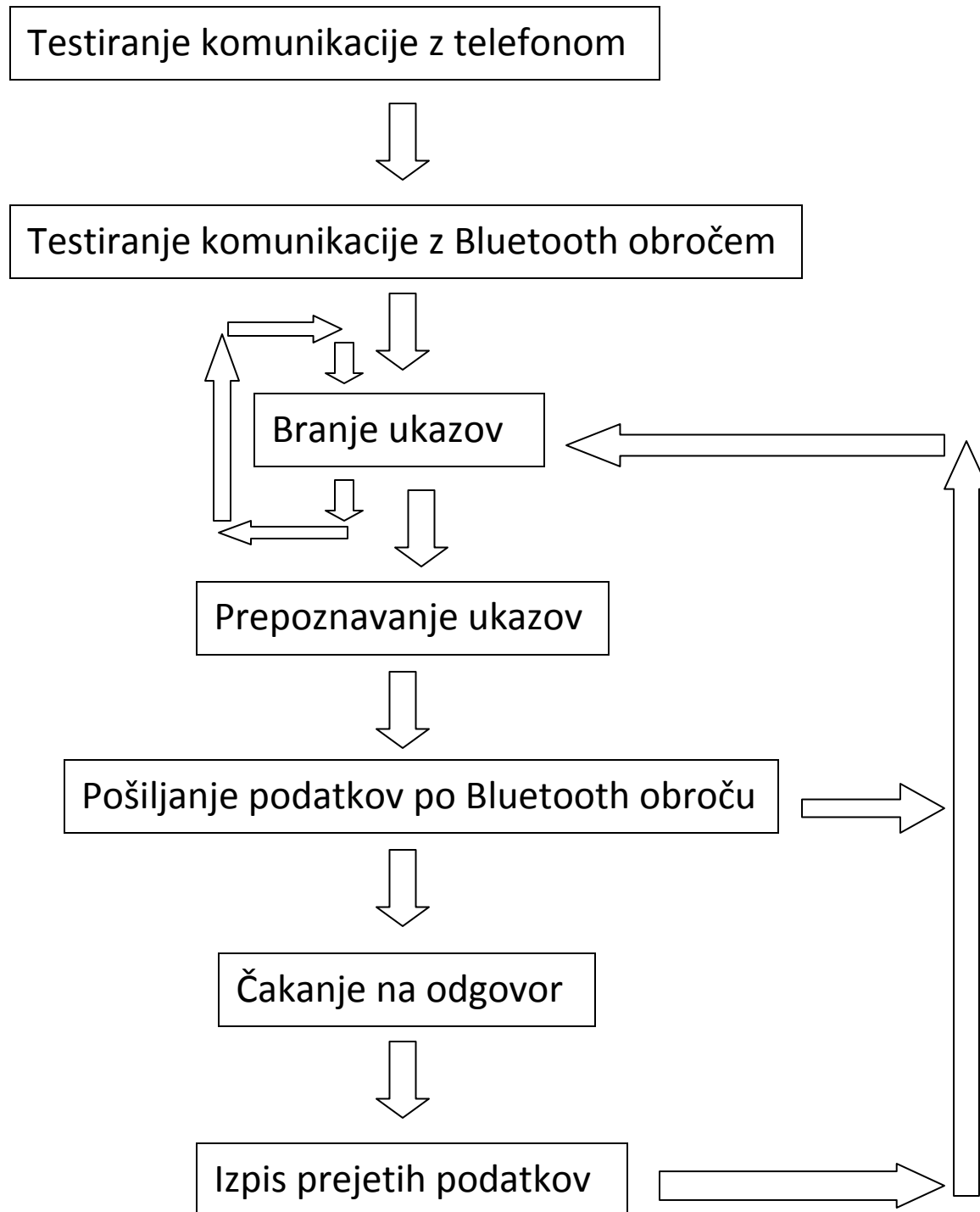
- sprejemati tudi ukaze iz telefona (uporabljam aplikacijo Bluetooth terminal HC05),
- te prejete ukaze iz telefona prepoznati,
- poslati ustrezno zaporedje ukazov v Bluetooth komunikacijsko mrežo,
- prebrati vse vračajoče podatke o temperaturah in osvetljenosti,

- izračunati povprečje vse treh prejetih podatkov senzorjev in
- jih poslati nazaj na telefon ter tako prikazati uporabniku.

Raspberry Pi prepozna naslednje ukaze:

- TMP? (prebere vrednosti vseh treh senzorjev v bluetooth zanki, izračuna povprečno vrednost, pretvori v °C in izpiše na telefonu in terminalu)
- SVET? (preberi vse tri vrednosti svetlobnih senzorjev, jih pošlji na telefon in izpiši na zaslonu)
- LUC1 (vklop luč)
- LUC0 (izklop luč)
- LUC? (preberi stanje luči in podatek pošlji telefonu)
- ŽALUD021 (krmili desno žaluzijo)
- ŽALUL038 (krmili levo žaluzijo)

Diagram poteka je prikazan na sliki 18. Koda programa se nahaja v prilogi.



Slika 18: Diagram poteka programa na osrednjem računalniku

### 3 Izdelava tiskanih vezij in ožičenje

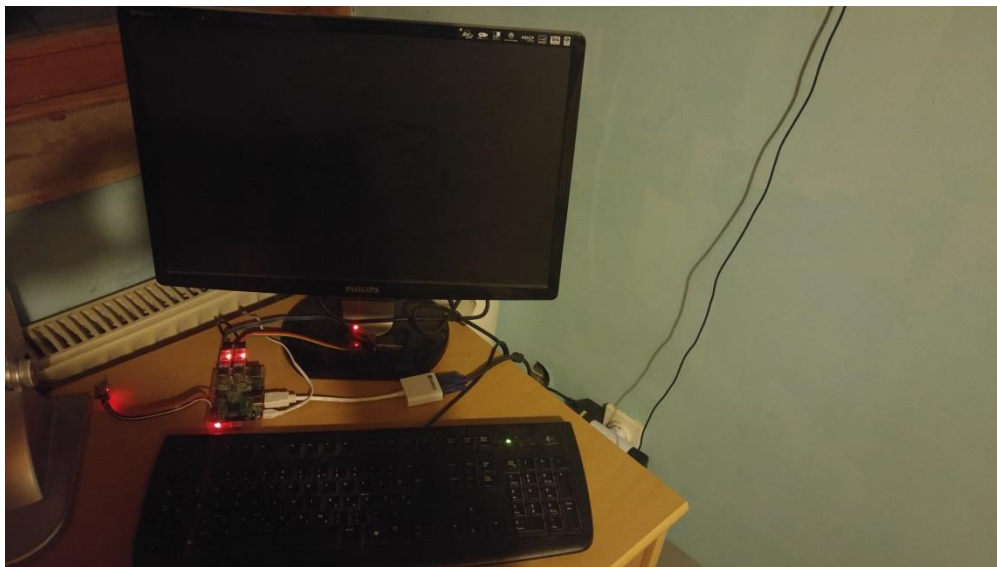
Postopki izdelave tiskanih vezij se niso niti malo spremenile, zato si lahko originalni opis teh postopkov preberete v prilogi.

V tem poglavju bo le opisano ožičenje, zagon ter konfiguriranje Raspberry Pi.

#### 3.1 Priprava Raspberry Pi

Predpogoj za zaganjanje Raspberry Pi računalnika je napajanje. Raspberry Pi ima priložen svoj napajalnik z dodano zmogljivostjo (2A), saj je računalnik potraten.

Raspberry Pi podpira HDMI in USB priključke, in se priklopi na ekran, miško in tipkovnico tako kot navadni osebni računalnik (PC).

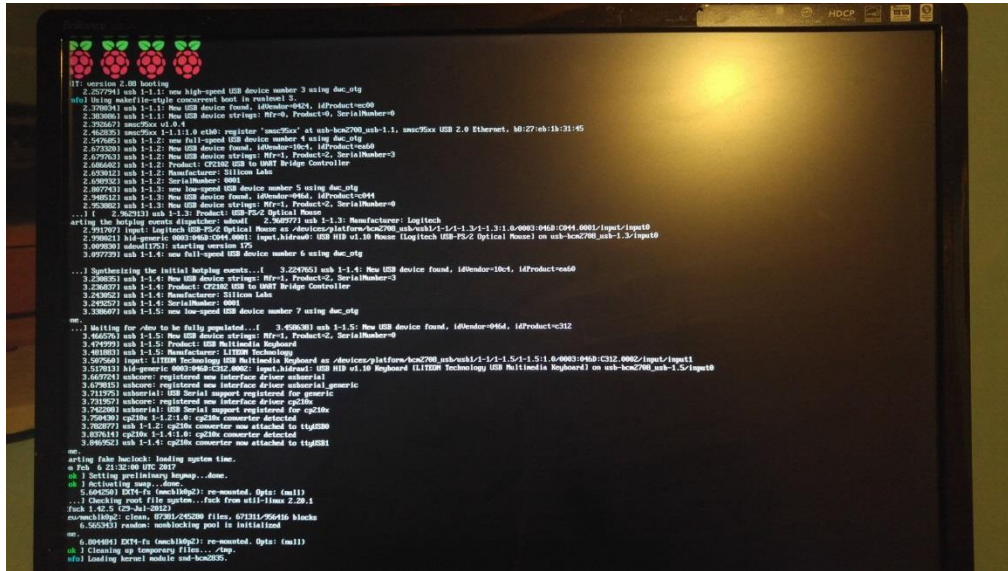


Slika 19: Prikaz celotnega priključenega sistema na Raspberry Pi

Iz slike 10 (popoglavje 2.1) in podpoglavja 2.2 je razvidno, da so potrebne na osrednjem računalniku 3 Bluetooth enote. Raspberry Pi ima le ena serijska vrata, zato sem uporabil za priklop ostalih Bluetooth enot USB vrata. Tako je ena Bluetooth enota povezana direktno na serijska vrata (AMA0), ostale Bluetooth enote pa so s pomočjo UART na TTL (USB) pretvornika povezane na USB vrata (USB0, USB1, USB2, USB3).

Zagon Raspberry Pi računalnika se začne z nalaganjem Raspbian [15] operacijskega sistema na mikro SD kartico. Priporočljivo je, da je SD kartica popolnoma prazna, saj lahko tudi manjša količina starih podatkov lahko pokvari delovanje operacijskega sistema (to vem iz izkušenj).

Ob vstavljeni SD kartici v Raspberry Pi računalnik se avtomatično začne zaganjati Raspberry Pi. Nato je potrebno nastaviti na računalniku osebne nastavitve kot so geslo, uporabniško ime, ravni pravic (uporabiške ali administrativne pravice),... omogočiti dostop do spleta, omogočiti dostop do terminala preko serijskega vhoda, ... .



### Slika 20. Prikaz Boot up procesa

Da lahko programiraš v zelenem okolju, si je potrebno to okolje naložiti. Kot sem že napisal v podpoglavju 2.7 o načrtovanju bom uporabljal Geany programsko okolje. Najlažji način za nalaganje programov na Raspberry Pi je ukaz `apt-get install`, ki ga vpišemo v terminal. Ukaz avtomatsko naloži vse datoteke in inštalira program. (potrebno je administrativno dovoljenje). Enako ponovimo, da naložimo knjižnico `WiringPi`.

### 3.2 Ožičenje

V mojem sistemu, kjer se poskušam izogniti vsemu ožičenju še vedno potrebujejo moje naprave 5V napajanje. Moja električna vezja porabijo preveč električnega toka, da bi bilo baterijsko napajanje praktično. Istočasno pa menim, da so baterije v takšnih sistemih slaba zamisel, saj jih je potrebno redno in pravočasno menjavati.

#### **4 Preizkušanje in odprava napak**

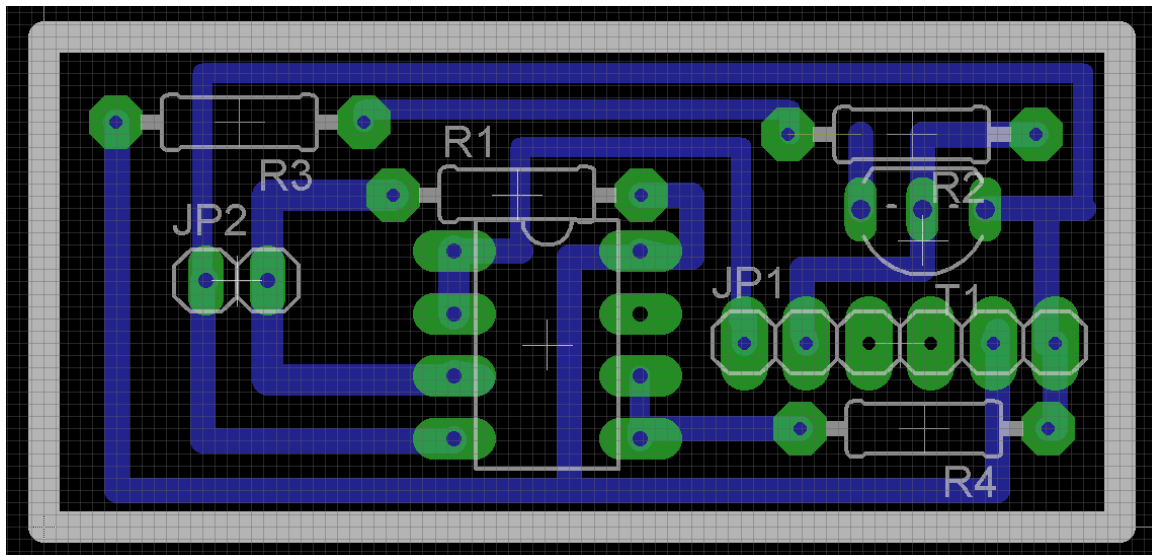
Pri gradnji velikih sistemov se vedno pojavijo napake. Te so lahko v napravah ali v programih. Oba vrsti napak je enako tečno odpravljati.

Podobno kot pri izdelavi tiskanih vezij ni razlike med sedanjim in preteklim odpravljanjem napak. V prilogi se nahaja opis najučinkovitejših postopkov odpravljanja napak tako kot tudi, kateri sistemi so namenjeni ugotavljanju napak.

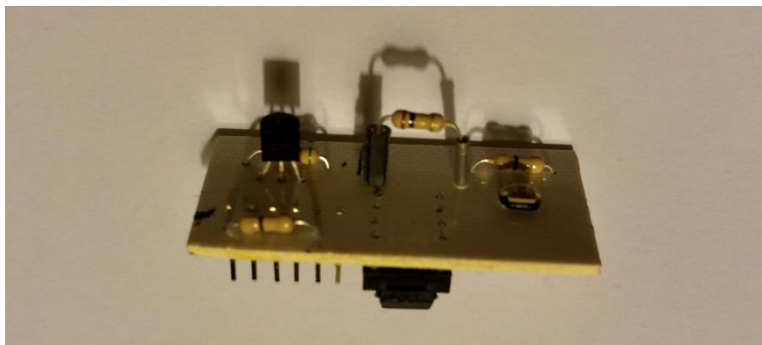
# Rezultati

Cilj vsakega dela je rezultat, saj drugače ne vem zakaj bi delali? V inovacijskem predlogu je končno čas za opis izdelka. Slike od 21 do 25 prikazujejo končan sistem Pametne hiše. Med izdelavo tega sistema sem naredil tudi eno malomarno napako, saj sem pri izdelavi tiskov le-te narobe zrcalil. Napaka ni bila usodna vendar mi je vseeno otežila delo.

Slika 21 prikazuje tiskano vezje svetlobnega in temperaturnega senzorja. Slika pod tem pa prikazuje izdelan senzor (z napako zrcaljenja).



Slika 21: Tiskano vezje svetlobnega in temperaturnega senzorja

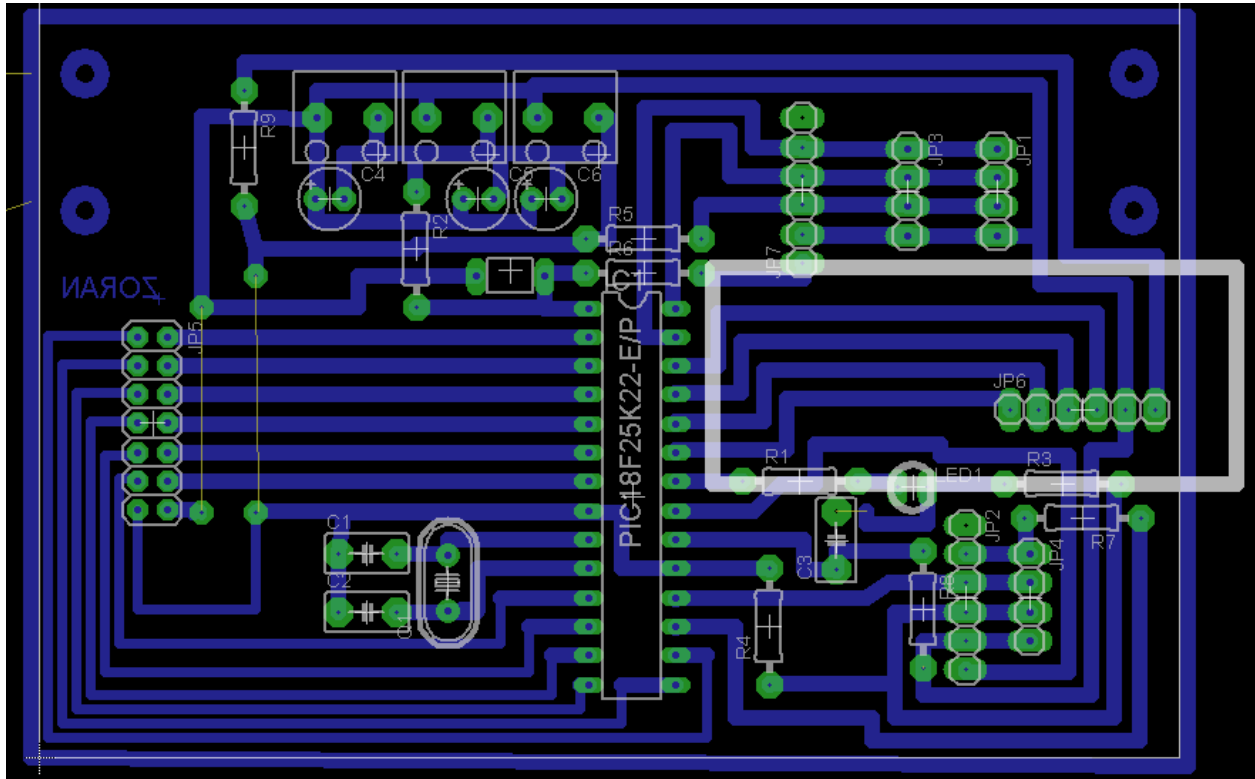


Slika 22: Izdelano vezje svetlobnega in temperaturnega senzorja

Nogice vidne od spodaj se vtaknejo v večje tiskano vezje mikrokrmilnika in tako povežejo obe vezji.

Tako lahko z »1-wire« protokolom beremo temperaturni senzor in z ukazom analog-read (oziroma popravljenim ekvivalentom) preberemo analogno vrednost svetlobnega senzorja.

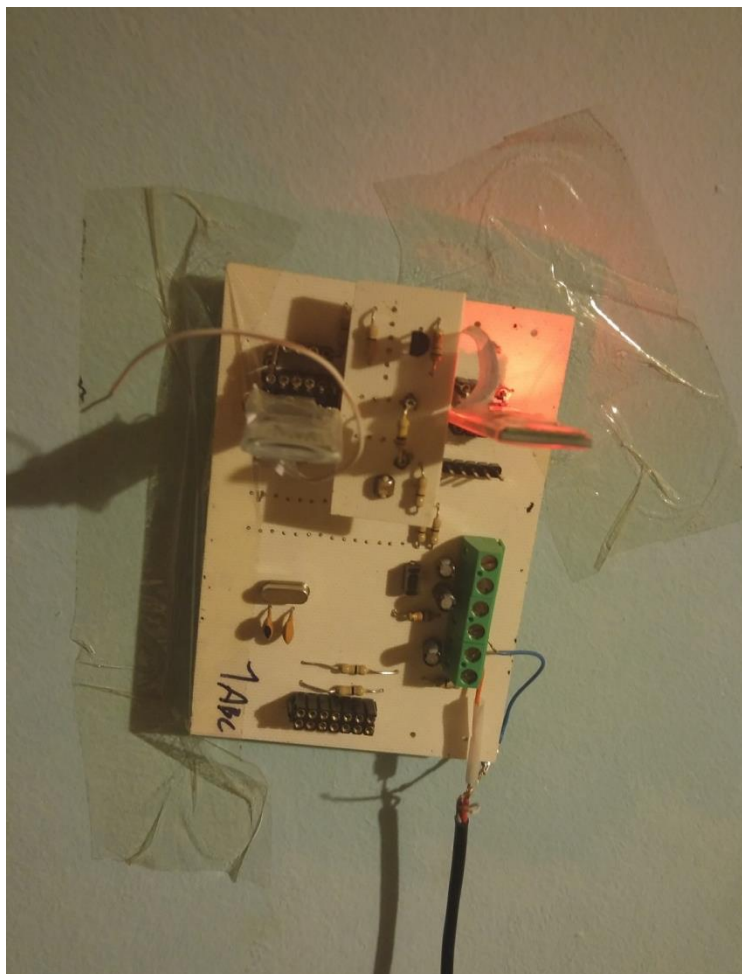
Slika 23 predstavlja končano ploščico tiskanega vezja mikrokrmilnika.



Slika 23: Tiskano vezje mikrokrmilnika

Na ploščici tiskanega vezja je krmilnik PIC18F25K22. Območje belega kvadra pa označuje območje, senzorske ploščice.

Na naslednji sliki pa vidimo izdelano ploščico tiskanega vezja mikrokrmilnika, pritrjeno na steno in opremljeno z svetlobnim senzorjem.

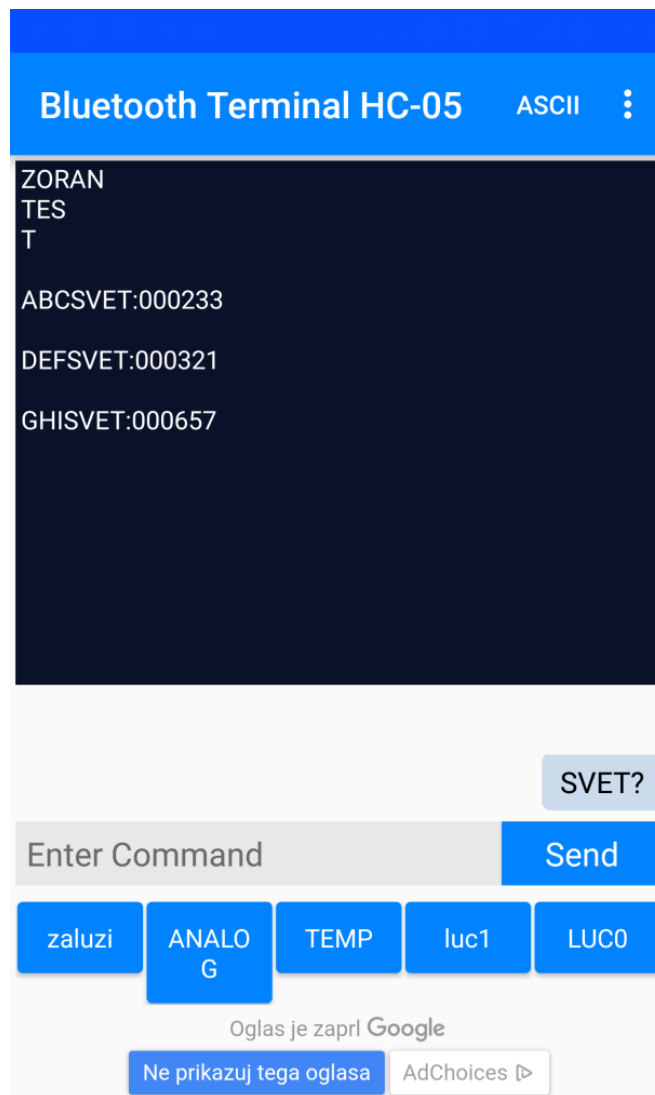


Slika 24: Izdelano in na steno pritrjeno procesorsko vezje

Metode pritrdjevanja so še nestalne in »primitivne«, zaradi napačnih meritev vmesniški ploščici, ki je bila namenjena pritrditvi.

To zaključuje moj tehnični del rezultatov. Programska koda se nahaja med prilogami, seznam ukazov na katere se odziva Raspberry Pi pa lahko najdemo v podpoglavju 2.7.

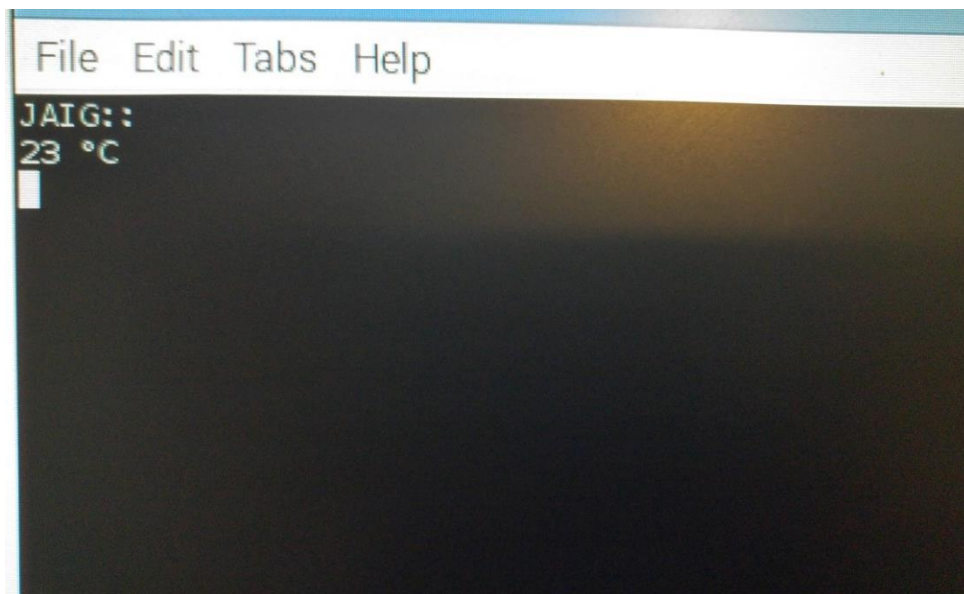
Na naslednji sliki pa vidimo delovanje aplikacije Bluetooth terminal HC-05[14], ki sem jo uporabljal za komuniciranje z mojimi Bluetooth napravami. Slika prikazuje zagon programa (in posledično testiranje vseh Bluetooth povezav) ter uspešno branje vseh treh svetlobnih senzorjev.



Slika 25: Prikaz komuniciranja z Raspberry Pi računalnikom

Izpis prikazuje nepretvorjene vrednosti, ki so bile prebrane na senzorjih in takoj poslane na moj telefon.

Program je napisan tako, da se vrednosti izpišejo tudi na zaslonu Raspberry Pi računalnika, kot prikazuje slika 26. V primeru temperature se izračuna povprečje vseh senzorjev in pretvori trimestno številko, ki predstavlja temperaturo (okoli 400) v °C.



Slika 26: Prikaz izpisa podatkov v terminalu na Raspberry Pi

## Razprava/Interpretacija rezultatov

Izboljšana verzija pametne sobe je pomagala uresničiti cilje, ki sem si jih zastavil že leta 2015. Moja soba je postala udobnejša kot kadarkoli prej. Sedaj imam lažji dostop do podatkov o svoji sobi in v bližnji prihodnosti ga bom uporabil za avtomatsko regulacijo.

Dostop do teh podatkov sicer še ni najbolj priročen. Vendar sta z uporabo računalnika Raspberry Pi obe pomanjkljivosti obsojeni na izumrtje.

Uspeh tukaj je še korak bližje (možni) popularizaciji takšnih pametnih inštalacij. To bo tudi omogočilo invalidom, starejšim ljudem dostop do takšnega sistema. Ta sistem ima za razliko od starega mnogo večjo možnost razširitve.

V primerjavi s podobnimi sistemi na tržišču pa ima prednost velike razširljivosti zaradi Bluetooth zanke in varnosti, ki jo zagotavlja sama tehnologija Bluetooth komuniciranja (preskakovanja frekvenc). Moj sistem Pametne hiše se lahko primerja z drugimi sistemi z brezžično inštalacijo. Večina takih sistemov krmili le luči, žaluzije in centralno ogevanje. Sicer imajo ti sistemi prednost zaradi večje stopnje integracije.

Verjamem, da je bil projekt izveden najboljše glede na moje tehnične in finančne zmožnosti ter omejeni čas, v katerem sem to naredil. Svoj projekt sem moral koordinirati tudi z drugimi projekti in obveznostmi.

# Zaključek

Ideja se ves čas izboljšuje. Izboljšave so bile vredne truda. Sicer nisem imel postavljenih zelo strogih ciljev, kot sem jih imel v prejšnji raziskovalni nalogi, vendar menim, da so bili cilji večinoma uresničeni (lahko bi lepše izdelal elektronske ploščice). Sistem ima skoraj neomejeno razširljivost in ga lahko razširim tudi na celo hišo (mogoče bo v prihodnosti ime spremenjeno). Menim, da bi sistem lahko dosti izboljšal z integracijo vezij za krmiljenje žaluzij in luči (torej releji). Te elemente bi vgradil direktno v doze in se tako izognil nadometnim škatlam. Imam tudi ideje glede robotov, priklopljenih na ta sistem. Odprtih je dosti novih smeri razvoja.

# Družbena odgovornost

Mislim, da je bilo moje delo družbeno odgovorno. Delal sem trdo in zavzeto. Z razvojem tega sistema želim izboljšati življenjski standard. Natančneje bi na dolgi rok rad povečal avtonomijo invalidom in starejšim ljudem. Tako kot vsi sistemi pametnih hiš z boljšo regulacijo temperature lahko pomeni večji prihranek energije, torej potroši manj fosilnih goriv za ogrevanje, kar pripomore k ohranjanju okolja. Vedno več čas se tudi zadržujemo v službi in ta sistem z določenimi nadgradnjami lahko prihrani dosti časa, ki bi bil drugače namenjen vzdrževanju gospodinjstva.

# Viri:

- [1] – Zoran Uran: Pametna soba, Mladi za napredek Maribora 2015, inovacijski predlog, OŠ Računalništvo; [http://zpm-mb.si/wp-content/uploads/2015/05/O%C5%A0\\_Ra%C4%8Dunalni%C5%A1tvo\\_IP\\_Pametna\\_soba.pdf](http://zpm-mb.si/wp-content/uploads/2015/05/O%C5%A0_Ra%C4%8Dunalni%C5%A1tvo_IP_Pametna_soba.pdf)
- [2] - [https://www.knx.org/media/docs/downloads/Marketing/Flyers/KNX-Basics/KNX-Basics\\_en.pdf](https://www.knx.org/media/docs/downloads/Marketing/Flyers/KNX-Basics/KNX-Basics_en.pdf)
- [3] - [https://en.wikipedia.org/wiki/Computer\\_network#Network\\_links](https://en.wikipedia.org/wiki/Computer_network#Network_links)  
- [https://sl.wikipedia.org/wiki/Ra%C4%8Dunalni%C5%A1ko\\_omre%C5%BEje](https://sl.wikipedia.org/wiki/Ra%C4%8Dunalni%C5%A1ko_omre%C5%BEje)
- [4] - <http://www.nutsvolts.com/magazine/article/light-sensitive-circuits>  
- <http://www.resistorguide.com/photoresistor/>  
- <https://en.wikipedia.org/wiki/Photoresistor>
- [5] - [http://www.gira.com/sl\\_SI/gebaeudetechnik/systeme/funk-bussystem/systemerlaeuterung.html](http://www.gira.com/sl_SI/gebaeudetechnik/systeme/funk-bussystem/systemerlaeuterung.html)  
- [http://www.gira.com/sl\\_SI/gebaeudetechnik/systeme/funk-bussystem/sender.html](http://www.gira.com/sl_SI/gebaeudetechnik/systeme/funk-bussystem/sender.html)
- [6] - <http://varcevanje-energije.si/inteligentne-instalacije/seltranova-aplikacija-clausius-in-kelvin.html>
- [7] - [http://www.kopp.eu/free-control-prospekt\\_web.pdf](http://www.kopp.eu/free-control-prospekt_web.pdf)  
- [http://www.kopp.eu/de/attachments/content/105286/systemubersicht-prospekt-3699.9938.4\\_april09.pdf](http://www.kopp.eu/de/attachments/content/105286/systemubersicht-prospekt-3699.9938.4_april09.pdf)  
- [http://www.kopp.eu/20\\_2015-datenblatt-mediola-gateway1.pdf](http://www.kopp.eu/20_2015-datenblatt-mediola-gateway1.pdf)
- [8] - [https://www.conrad.si/Brezzicni-termostat-eQ-3-CC-RT-BLE.htm?websale8=conrad-slowenien&pi=1364875&ci=SHOP\\_AREA\\_37360\\_0812040](https://www.conrad.si/Brezzicni-termostat-eQ-3-CC-RT-BLE.htm?websale8=conrad-slowenien&pi=1364875&ci=SHOP_AREA_37360_0812040)  
- <https://www.conrad.si/Max%21-radiatorski-termostat-eQ-3-Max%21-Plus,-bele-barve-105936.htm?websale8=conrad-slowenien&pi=519588>
- [9] - <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>  
- <http://www.raspberrypi-spy.co.uk/2012/06/simple-guide-to-the-rpi-gpio-header-and-pins/>

- [10] - <http://wiringpi.com/>  
- <http://wiringpi.com/reference/serial-library/>
- [11] - <http://sl.wikipedia.org/wiki/Radio>  
- [http://en.wikipedia.org/wiki/Radio\\_frequency](http://en.wikipedia.org/wiki/Radio_frequency)  
- <http://www.ni.com/tutorial/3541/en/>
- [12] - <http://www.informit.com/articles/article.aspx?p=21324&seqNum=4>  
- <http://blog.zakkemble.co.uk/getting-bluetooth-modules-talking-to-each-other/>  
- [http://en.wikipedia.org/wiki/MAC\\_address](http://en.wikipedia.org/wiki/MAC_address)  
- <https://learn.sparkfun.com/tutorials/bluetooth-basics/bluetooth-profiles>  
- <https://learn.sparkfun.com/tutorials/bluetooth-basics/how-bluetooth-works>  
- <http://phillipecantin.blogspot.com/2014/01/hc-05-bluetooth-link-of-2-arduino.html>  
- <http://www.instructables.com/id/BlueTooth-Link-with-auto-detect-connect/?ALLSTEPS>
- [13] - <http://www.amarino-toolkit.net/>
- [14] - <https://play.google.com/store/apps/details?id=project.bluetoothterminal&hl=sl>
- [15] - <https://www.raspberrypi.org/downloads/raspbian/>

#### **Dotatni viri (prebrana literatura):**

- S.Monk: Arduino+Android projects for the evil genius, McGraw-Hill, New York, Chicago, London, Madrid, 2012
- Daniel Černi, Digitalni sistem za merjenje in brezžični prenos temperaturnih podatkov <https://dk.um.si/Dokument.php?id=50291>
- Andrej Perko, Načrtovanje in izvedba pametne hiše z mikrokrmilnikom Arduino <https://dk.um.si/Dokument.php?id=82065>
- Dušan Leskovar, Projektiranje električnih inštalacij v inteligentnih zgradbah <https://dk.um.si/Dokument.php?id=8080>

# Viri slike in razpredelnice:

- Slika 1: [http://2.bp.blogspot.com/-jvixH9FQ9TI/UiO5ObAtm5I/AAAAAAAAAGc/FCYy5O\\_48nw/s1600/connection.jpg](http://2.bp.blogspot.com/-jvixH9FQ9TI/UiO5ObAtm5I/AAAAAAAAAGc/FCYy5O_48nw/s1600/connection.jpg)
- Slika 2: [http://mcuoneclipse.files.wordpress.com/2013/06/jy-mcu-bt\\_board-v1-05-bottom-side.png?w=584&h=275](http://mcuoneclipse.files.wordpress.com/2013/06/jy-mcu-bt_board-v1-05-bottom-side.png?w=584&h=275)
- Slika 3: <http://www.seminarski-diplomski.co.rs/INFORMATIKA/pictures/Topologija%20scatternet-a.JPG>
- Slika 4: Avtor
- Slika 5: Avtor
- Slika 6: Avtor
- Slika 7: Avtor
- Slika 8: Avtor
- Slika 9: Avtor
- Slika 10: Avtor
- Slika 11: Avtor
- Slika 12: Avtor
- Slika 13: Avtor
- Slika 14: Avtor
- Slika 15: Avtor
- Slika 16: Avtor
- Slika 17: Avtor
- Slika 18: Avtor
- Slika 19: Avtor
- Slika 20: Avtor
- Slika 21: Avtor
- Slika 22: Avtor
- Slika 23: Avtor
- Slika 24: Avtor
- Slika 25: Avtor
- Slika 26: Avtor
- Slika 27: Avtor
- Slika 28: Avtor
- Slika 29: Avtor
- Slika 30: Avtor

- Slika 31: Avtor
- Slika 32: Avtor
- Slika 33: Avtor
- Slika 34: Avtor
- Slika 35:  
[https://upload.wikimedia.org/wikipedia/commons/3/32/Topolox%C3%ADa\\_en\\_anel.png](https://upload.wikimedia.org/wikipedia/commons/3/32/Topolox%C3%ADa_en_anel.png)
- Slika 36:  
[https://upload.wikimedia.org/wikipedia/commons/1/1c/Topolox%C3%ADa\\_en\\_estrela.png](https://upload.wikimedia.org/wikipedia/commons/1/1c/Topolox%C3%ADa_en_estrela.png)
- Slika 37:  
[https://upload.wikimedia.org/wikipedia/commons/3/30/Topolox%C3%ADa\\_en\\_bus.png](https://upload.wikimedia.org/wikipedia/commons/3/30/Topolox%C3%ADa_en_bus.png)
- Slika 38:  
[https://upload.wikimedia.org/wikipedia/commons/8/85/Topolox%C3%ADa\\_en\\_malla\\_completa.png](https://upload.wikimedia.org/wikipedia/commons/8/85/Topolox%C3%ADa_en_malla_completa.png)
- Slika 39:  
[https://upload.wikimedia.org/wikipedia/commons/5/5a/Topolox%C3%ADa\\_en\\_%C3%A1rbore.png](https://upload.wikimedia.org/wikipedia/commons/5/5a/Topolox%C3%ADa_en_%C3%A1rbore.png)
- Slika 40:  
[https://upload.wikimedia.org/wikipedia/commons/5/55/Topolox%C3%ADa\\_en\\_rede\\_celular.png](https://upload.wikimedia.org/wikipedia/commons/5/55/Topolox%C3%ADa_en_rede_celular.png)
- Razpredelnica 1: Avtor
- Razpredelnica 2: Avtor

# Priloge

## Priloga A: Ascii tabela

| Dec | Hx | Oct | Char                               | Dec | Hx | Oct | Html  | Chr          | Dec | Hx | Oct | Html  | Chr      | Dec | Hx | Oct | Html   | Chr        |
|-----|----|-----|------------------------------------|-----|----|-----|-------|--------------|-----|----|-----|-------|----------|-----|----|-----|--------|------------|
| 0   | 0  | 000 | <b>NUL</b> (null)                  | 32  | 20 | 040 | &#32; | <b>Space</b> | 64  | 40 | 100 | &#64; | <b>@</b> | 96  | 60 | 140 | &#96;  | <b>`</b>   |
| 1   | 1  | 001 | <b>SOH</b> (start of heading)      | 33  | 21 | 041 | &#33; | <b>!</b>     | 65  | 41 | 101 | &#65; | <b>A</b> | 97  | 61 | 141 | &#97;  | <b>a</b>   |
| 2   | 2  | 002 | <b>STX</b> (start of text)         | 34  | 22 | 042 | &#34; | <b>"</b>     | 66  | 42 | 102 | &#66; | <b>B</b> | 98  | 62 | 142 | &#98;  | <b>b</b>   |
| 3   | 3  | 003 | <b>ETX</b> (end of text)           | 35  | 23 | 043 | &#35; | <b>#</b>     | 67  | 43 | 103 | &#67; | <b>C</b> | 99  | 63 | 143 | &#99;  | <b>c</b>   |
| 4   | 4  | 004 | <b>EOT</b> (end of transmission)   | 36  | 24 | 044 | &#36; | <b>\$</b>    | 68  | 44 | 104 | &#68; | <b>D</b> | 100 | 64 | 144 | &#100; | <b>d</b>   |
| 5   | 5  | 005 | <b>ENQ</b> (enquiry)               | 37  | 25 | 045 | &#37; | <b>%</b>     | 69  | 45 | 105 | &#69; | <b>E</b> | 101 | 65 | 145 | &#101; | <b>e</b>   |
| 6   | 6  | 006 | <b>ACK</b> (acknowledge)           | 38  | 26 | 046 | &#38; | <b>&amp;</b> | 70  | 46 | 106 | &#70; | <b>F</b> | 102 | 66 | 146 | &#102; | <b>f</b>   |
| 7   | 7  | 007 | <b>BEL</b> (bell)                  | 39  | 27 | 047 | &#39; | <b>'</b>     | 71  | 47 | 107 | &#71; | <b>G</b> | 103 | 67 | 147 | &#103; | <b>g</b>   |
| 8   | 8  | 010 | <b>BS</b> (backspace)              | 40  | 28 | 050 | &#40; | <b>(</b>     | 72  | 48 | 110 | &#72; | <b>H</b> | 104 | 68 | 150 | &#104; | <b>h</b>   |
| 9   | 9  | 011 | <b>TAB</b> (horizontal tab)        | 41  | 29 | 051 | &#41; | <b>)</b>     | 73  | 49 | 111 | &#73; | <b>I</b> | 105 | 69 | 151 | &#105; | <b>i</b>   |
| 10  | A  | 012 | <b>LF</b> (NL line feed, new line) | 42  | 2A | 052 | &#42; | <b>*</b>     | 74  | 4A | 112 | &#74; | <b>J</b> | 106 | 6A | 152 | &#106; | <b>j</b>   |
| 11  | B  | 013 | <b>VT</b> (vertical tab)           | 43  | 2B | 053 | &#43; | <b>+</b>     | 75  | 4B | 113 | &#75; | <b>K</b> | 107 | 6B | 153 | &#107; | <b>k</b>   |
| 12  | C  | 014 | <b>FF</b> (NP form feed, new page) | 44  | 2C | 054 | &#44; | <b>,</b>     | 76  | 4C | 114 | &#76; | <b>L</b> | 108 | 6C | 154 | &#108; | <b>l</b>   |
| 13  | D  | 015 | <b>CR</b> (carriage return)        | 45  | 2D | 055 | &#45; | <b>-</b>     | 77  | 4D | 115 | &#77; | <b>M</b> | 109 | 6D | 155 | &#109; | <b>m</b>   |
| 14  | E  | 016 | <b>SO</b> (shift out)              | 46  | 2E | 056 | &#46; | <b>.</b>     | 78  | 4E | 116 | &#78; | <b>N</b> | 110 | 6E | 156 | &#110; | <b>n</b>   |
| 15  | F  | 017 | <b>SI</b> (shift in)               | 47  | 2F | 057 | &#47; | <b>/</b>     | 79  | 4F | 117 | &#79; | <b>O</b> | 111 | 6F | 157 | &#111; | <b>o</b>   |
| 16  | 10 | 020 | <b>DLE</b> (data link escape)      | 48  | 30 | 060 | &#48; | <b>0</b>     | 80  | 50 | 120 | &#80; | <b>P</b> | 112 | 70 | 160 | &#112; | <b>p</b>   |
| 17  | 11 | 021 | <b>DC1</b> (device control 1)      | 49  | 31 | 061 | &#49; | <b>1</b>     | 81  | 51 | 121 | &#81; | <b>Q</b> | 113 | 71 | 161 | &#113; | <b>q</b>   |
| 18  | 12 | 022 | <b>DC2</b> (device control 2)      | 50  | 32 | 062 | &#50; | <b>2</b>     | 82  | 52 | 122 | &#82; | <b>R</b> | 114 | 72 | 162 | &#114; | <b>r</b>   |
| 19  | 13 | 023 | <b>DC3</b> (device control 3)      | 51  | 33 | 063 | &#51; | <b>3</b>     | 83  | 53 | 123 | &#83; | <b>S</b> | 115 | 73 | 163 | &#115; | <b>s</b>   |
| 20  | 14 | 024 | <b>DC4</b> (device control 4)      | 52  | 34 | 064 | &#52; | <b>4</b>     | 84  | 54 | 124 | &#84; | <b>T</b> | 116 | 74 | 164 | &#116; | <b>t</b>   |
| 21  | 15 | 025 | <b>NAK</b> (negative acknowledge)  | 53  | 35 | 065 | &#53; | <b>5</b>     | 85  | 55 | 125 | &#85; | <b>U</b> | 117 | 75 | 165 | &#117; | <b>u</b>   |
| 22  | 16 | 026 | <b>SYN</b> (synchronous idle)      | 54  | 36 | 066 | &#54; | <b>6</b>     | 86  | 56 | 126 | &#86; | <b>V</b> | 118 | 76 | 166 | &#118; | <b>v</b>   |
| 23  | 17 | 027 | <b>ETB</b> (end of trans. block)   | 55  | 37 | 067 | &#55; | <b>7</b>     | 87  | 57 | 127 | &#87; | <b>W</b> | 119 | 77 | 167 | &#119; | <b>w</b>   |
| 24  | 18 | 030 | <b>CAN</b> (cancel)                | 56  | 38 | 070 | &#56; | <b>8</b>     | 88  | 58 | 130 | &#88; | <b>X</b> | 120 | 78 | 170 | &#120; | <b>x</b>   |
| 25  | 19 | 031 | <b>EM</b> (end of medium)          | 57  | 39 | 071 | &#57; | <b>9</b>     | 89  | 59 | 131 | &#89; | <b>Y</b> | 121 | 79 | 171 | &#121; | <b>y</b>   |
| 26  | 1A | 032 | <b>SUB</b> (substitute)            | 58  | 3A | 072 | &#58; | <b>:</b>     | 90  | 5A | 132 | &#90; | <b>Z</b> | 122 | 7A | 172 | &#122; | <b>z</b>   |
| 27  | 1B | 033 | <b>ESC</b> (escape)                | 59  | 3B | 073 | &#59; | <b>;</b>     | 91  | 5B | 133 | &#91; | <b>[</b> | 123 | 7B | 173 | &#123; | <b>{</b>   |
| 28  | 1C | 034 | <b>FS</b> (file separator)         | 60  | 3C | 074 | &#60; | <b>&lt;</b>  | 92  | 5C | 134 | &#92; | <b>\</b> | 124 | 7C | 174 | &#124; | <b> </b>   |
| 29  | 1D | 035 | <b>GS</b> (group separator)        | 61  | 3D | 075 | &#61; | <b>=</b>     | 93  | 5D | 135 | &#93; | <b>]</b> | 125 | 7D | 175 | &#125; | <b>}</b>   |
| 30  | 1E | 036 | <b>RS</b> (record separator)       | 62  | 3E | 076 | &#62; | <b>&gt;</b>  | 94  | 5E | 136 | &#94; | <b>^</b> | 126 | 7E | 176 | &#126; | <b>~</b>   |
| 31  | 1F | 037 | <b>US</b> (unit separator)         | 63  | 3F | 077 | &#63; | <b>?</b>     | 95  | 5F | 137 | &#95; | <b>_</b> | 127 | 7F | 177 | &#127; | <b>DEL</b> |

Source: [www.LookupTables.com](http://www.LookupTables.com)

Slika 27

#### Priloga B: Tabela uporabljenih ukazov v poveljniškem načinu Bluetooth enot

|                                      |                                                                                                        |
|--------------------------------------|--------------------------------------------------------------------------------------------------------|
| AT                                   | Preizkus delovanja serijske komunikacije.                                                              |
| AT+INQ                               | Preveri, če so v dosegu druge bluetooth naprave                                                        |
| AT+INIT                              | Inicializira »Bluetooth profil« in tako omogoča povezavo z drugimi Bluetooth napravami                 |
| AT+PAIR=<naslov naprave>; <dani čas> | Prva povezava. Pri katerem si Bluetooth enoti izmenjata podatke, da se bosta lažje povezala naslednjič |
| AT+LINK=<naslov naprave>             | Vzpostavi povezavo med dvema Bluetooth napravama                                                       |
| AT+DISC                              | Prekine povezavo med dvema Bluetooth napravama                                                         |
| AT+CMODE=<stanje>                    | Vklopi ali izklopi avtomatsko povezovanje z vsemi drugimi Bluetooth napravami                          |
| AT+BIND=<naslov naprave>             | Bluetooth enota se avtomatsko poveže z določeno Bluetooth enoto                                        |

Razpredelnica 2

## PRILOGA C

### C1. Opis izdelave elektronskih tiskov

#### C1.1 Izdelava elektronskega vezja (jedkanje, vrtanje)

Proces izdelovanja ploščice se začne s tiskanjem že nastalega načrta na poseben svileni papir (v tem primeru na svileni papir reklam). Med tiskanjem vzamemo v roke vitroplast (pobakrena na eni strani) in jo pobrusim z gladkim brusnim papirjem, da odstranim zamaščen baker (pri tem tudi pomaga detergent).



Slika 28: Brušenje

Pred kratkim natisnjen papir obrežemo na velikost vitroplasta in ga spustimo skozi laminator. Barva na papirju se stopi in zalepi na pobakreni del vitroplasta.



Slika 29: Vitroplast ovit v svilen papir (reklamo)

Skozi laminator vitroplast spuščamo približno 15 minut.



Slika 30: Segravanje ploščic

Nato pa vitroplast (z zdaj prilepljenim papirjem) potopimo v mešanico 20% solne kisline in vodikovega peroksida (razmerje 10 : 1).



Slika 31: Kislina in vodikov peroksid

Baker takoj reagira s kislino in se začne raztapljati.



Slika 32: jedkanje 1

Kislino mešamo, dokler ne ostane samo načrt (in baker pod njem).

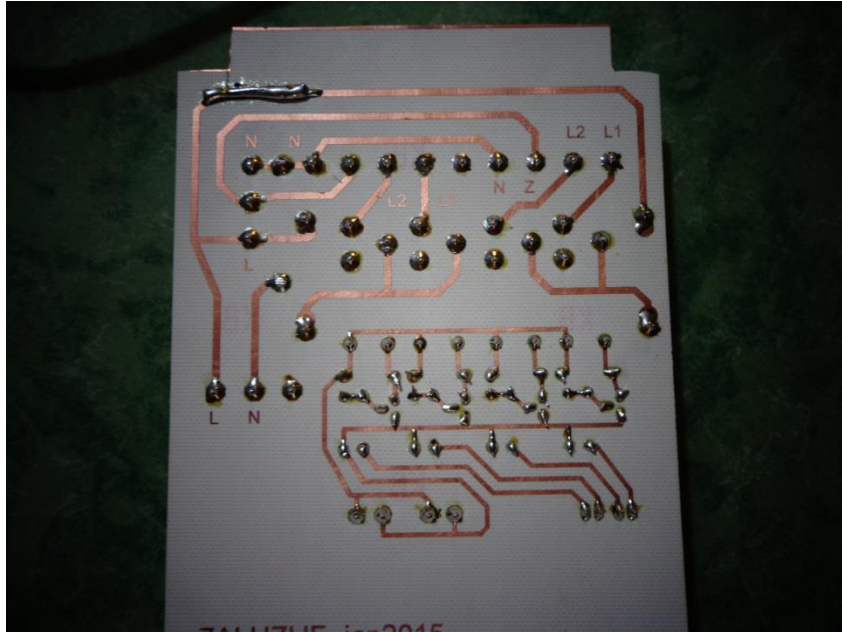


Slika 33: jedkanje 2

Po končanem delu vzamemo vitroplast iz kisline, ga operemo z vodo in z acetonom počistimo odvečno barvo. Na koncu še samo zvrtamo luknje. Po potrebi popravim kakšno napako, sicer je elektronsko vezje končano.

## C1.2 Spajkanje

Proces spajkanja se začne z zbiranjem potrebnih električnih (elektronskih) elementov. V mojem primeru sem za prejemniško vezje potreboval pic18f14k50, 40 pinsko podnožje, 5 sponk, 2 10nF kondenzatorja, 1 100  $\mu$ F kondenzator, 2 10  $\mu$ F kondenzatorja, 1 4MHz quartz, 1 LED, 1 1k $\Omega$  in 1 10k $\Omega$  upor. Ko so ti elementi zbrani, segrejemo spajkalnik in pripravimo orodje. V luknje vstavimo elektronske elemente (najnižji k najvišjim) in jih sproti spajkamo. To nadaljujemo, dokler niso vsi elementi prispajkani.



Slika 34: Zaspajkan tiskano vezje (pogled od spodaj)

Ko končamo, preverimo svoje delo, npr. če ima mikroprocesor napajanje.

## C2. Opis programiranja

S končanimi in preizkušenimi ploščicami mikrokrmilnika sem se osredotočil na programski del inovacijskega predloga. Programiral sem v programskem jeziku C. Osnova programiranja v tem jeziku je vpisovanje funkcij matematičnih peracij in »krmilnih« (if,While,else) stavkov. V naslednjem primeru sem uporabljal knjižnico funkcij za serijsko komunikacijo (npr. UART\_init(9600)).

Primer programa:

```
while (Lastbyte2 != ACK) // Tu preveri če zadnji byte ki ga je prebral ni ascii znak ACK
{
 while (UART1_Data_Ready() == 1) // Tu preveri če je naslednji byte pripravljen na branje
 {
 Lastbyte2 = UART1_Read(); // Tukaj prebere UART port in ga shrani v spremenljivko Lastbyte2
 buffer2[buffercount2] = Lastbyte2; // Vrednost Lastbyte2 prepíše v buffer z številko buffercount2
 buffercount2 = buffercount2 + 1; // buffercount2 se prišteje 1
 }
}
```

Slika 35: Program za branje serijske komunikacije

Na sliki 35 vidimo del programa za branje niza znakov s serijskih vrat. Program se ponavlja in shrani prežete znake v vmesnik z imenom buffer2, dokler ne prebere ascii znak ACK (Acknowledge) z desetiško vrednostjo 19.

Naslednji primer (slika 36) prikazuje izsek programa za branje podatkov, razčlenitev podatkov in tudi izvedbo naloge (vklop ali izklop luči).

```

if (Prvibyte == 0x41) // če je prvibyte enak A
{
 if (buffer[0] == 0x4C) // če je buffer[0] enak L
 {
 if (buffer[1] == 0x55) // če je buffer[1] enak U
 {
 if (buffer[2] == 0x31) // če je buffer[2] enak 1
 {
 if (PORTC.B5 == 1) // če je luč vklopljena
 {
 while (UART1_Tx_Idle == 0); //ali so serijska vrata prosta za pošiljanje podatkov
 UART1_Write(0x4F); // Pošlji ascii znak O
 while (UART1_Tx_Idle == 0); //ali so serijska vrata prosta za pošiljanje podatkov
 UART1_Write(0x4B); // pošlji ascii znak K
 while (UART1_Tx_Idle == 0); // ali so serijska vrata prosta za pošiljanje podatkov
 UART1_Write(19); // pošlji ascii znak ACK
 if (stanje == 0) // če luč ni vklopljena
 {
 PORTC = 0b00011000; // vklopi luč
 stanje = stanje + 1; // spremenljivko stanje povečaj za 1
 }
 else // če je luč vklopljena
 {
 PORTC = 0b00000000; // izklopi luč
 stanje = stanje - 1; // stanje pomanjšaj za 1
 }
 }
 }
 }
 }
}

```

Slika 36: Primer programa za krmiljenje luči

### C3. Program na mikrokrmilniku vsake naprave:

```
const unsigned short TEMP_RESOLUTION = 12;
unsigned temp = 0;
char UART1T[20];
char UART1R[20];
char UART2T[20];
char UART2R[20];
int ukaz = 0;
int stetje1 = 0;
int stetje2 = 0;
int brisanje = 0;
int prepisovanje = 0;
char crka1 = 0;
char crka2 = 0;
double tempC = 0;
char tempCZ[10];
int light = 0;
int kljuc = 0;
int Alrdy = 0;
```

```
void UART1T_reset()
{ brisanje = 0;
while (brisanje < 21)
{
 UART1T[brisanje] = 0;
 brisanje++;
}
brisanje = 0;
}
```

```
void UART2T_reset()
{ brisanje = 0;
while (brisanje < 21)
{
 UART2T[brisanje] = 0;
 brisanje++;
}
```

---

```

 }
 brisanje = 0;

}

void UART1R_reset()
{ brisanje = 0;

while (brisanje < 21)
{
 UART1R[brisanje] = 0;
 brisanje++;
}
 brisanje = 0;

}

void UART2R_reset()
{ brisanje = 0;
while (brisanje < 21)
{
 UART2R[brisanje] = 0;
 brisanje++;

}
 brisanje = 0;
}

void main()
{

ANSELC = 0;
ANSELB = 0b00000100;
TRISB = 0b00001100;
TRISC = 0;
ANSELA = 0; //
TRISA = 0; //
PORTB = 0b00000000;

```

---

```

// inicializacija A/D pretvorbe
ADCON1=0; // VDD in VSS sta Vref ta A/D, trigger = CMTU
ADCON2=0xB5; // 10-bitna desna poravnava rezultata, 16 TAD, FOSC/16
ADCON0=0x21; // AN8 -> CHS = 01000 , vklop A/D ADON =1
delay_us(100);
delay_ms(100);
UART2_init(9600);
delay_ms(100);
UART1_init(9600);
delay_ms(100);
UART1T_reset();
UART2T_reset();
UART1R_reset();
UART2R_reset();
PORTB = 0b00000010; // utripanje LED
delay_ms(1000);
PORTB = 0b00000000;
delay_ms(1000);

```

```

while (1)
{

if(UART1_data_ready() == 1)
{
while (crka1 != 0x0A) //BRanje UART1 !!!! NISI prebral
{
while (UART1_data_ready() != 1);
crka1 = UART1_read();
UART1R[stetje1] = crka1;
stetje1++;

}

stetje1 = 0;
crka1 = 0;

```

---

```
}
```

```
if(UART2_data_ready() == 1)
{
 while (crka2 != 0x0A) //BRanje UART2
 {
 while (UART2_data_ready() != 1);
 crka2 = UART2_read();
 UART2R[stetje2] = crka2;
 stetje2++;
 }
 stetje2 = 0;
 crka2 = 0;
}
```

```
////////////////////////////////////
```

```
if (UART2R[0] == 65 && UART2R[1] == 66 && UART2R[2] == 67 /*&& kljuc == 3*/)
{
/* while (prepisovanje < 21)
{
 UART2T[prepisovanje] = UART2R[prepisovanje];
 prepisovanje++;
}
 prepisovanje = 0; */
 ukaz = 1;

}
else if (UART2R[0] != 0 /*&& kljuc == 3*/)
{
 while (prepisovanje < 21)
 {
 UART1T[prepisovanje] = UART2R[prepisovanje];
 prepisovanje++;
 }
 prepisovanje = 0;
}
```

---

```

 UART2R_reset();
}
////////////////////////////////////

if (UART1T[0] != 0)
{
 while (UART1_tx_idle() != 1);
 UART1_write_text(UART1T);
 UART1T_reset();
}
if (UART2T[0] != 0)
{
 while (UART2_tx_idle() != 1);
 UART2_write_text(UART2T);
 UART2T_reset();
}
////////////////////////////////////

if (ukaz == 1)
{
 if(UART2R[3] == 65 && UART2R[4] == 78 && UART2R[5] == 63)
 {

 ADCON0.GO = 1; // start A/D pretvorbe
 while(ADCON0.GO); // čakaj na konec A/D pretvorbe
 light = ADRESH*256+ADRESL; // preberi rezultat v integer
 IntToStrWithZeros(light, UART1T);
 while (UART1_tx_idle() != 1);
 UART1_write_text ("ABCSVET:");
 UART1_write_text (UART1T);
 UART1_Write(0x0D);
 UART1_Write(0x0A);
 ANSELB = 0b00000100;
 TRISB = 0b00001100;
 UART1T_reset();
 }
 if(UART2R[3] == 84 && UART2R[4] == 77 && UART2R[5] == 80 && UART2R[6] == 63)
 {
 Ow_Reset(&PORTB, 3); // Onewire reset signal
 }
}

```

---

```

Ow_Write(&PORTB, 3, 0xCC); // Issue command SKIP_ROM
Ow_Write(&PORTB, 3, 0x44); // Issue command CONVERT_T
Delay_us(120);

Ow_Reset(&PORTB, 3);
Ow_Write(&PORTB, 3, 0xCC); // Issue command SKIP_ROM
Ow_Write(&PORTB, 3, 0xBE); // Issue command READ_SCRATCHPAD

temp = Ow_Read(&PORTB, 3);
temp = (Ow_Read(&PORTB, 3) << 8) + temp; // izmerjena temperatura
IntToStrwithzeros(temp, UART1T); // pretvorba v ascii
while (UART1_tx_idle() != 1);
UART1_write_text ("ABCTEMP:");
UART1_write_text (UART1T);
UART1_Write(0x0D);
UART1_Write(0x0A);
UART1T_reset();
}
UART2R_reset();
ukaz = 0;
}
}
}

```

---

## C4. Program na Raspberry Pi računalniku

```
#include <stdio.h> // potrebujem za printf
#include <wiringPi.h>
#include <wiringSerial.h>
const int ledPin = 23;
int S1_podatek = 0;
int M2_podatek = 0;
int ST3_podatek = 0;
int branje = 0;
int stevilo = 0;
char podatki[100];
char S1[100];
char M2[100];
char ST3[100];
int S1pretvorba = 0;
int S2pretvorba = 0;
int S3pretvorba = 0;
int S4pretvorba = 0;
int K1pretvorba = 0;
int K2pretvorba = 0;
int K3pretvorba = 0;
int Kpretvorba = 0;
int SP1 = 1;
int SP2 = 0;
int SP3 = 0;
int enkrat = 0;
int logika = 0;
int logika2 = 0;
void UART_Tx_reset(void)
{
 while (branje < 100)
 {
 podatki[branje] = 0;
 branje++;
 }
 branje = 0;
}
void UART_S1_reset(void)
{
 while (branje < 100)
 {
 S1[branje] = 0;
 branje++;
 }
 branje = 0;
}
void UART_M2_reset(void)
{
 while (branje < 100)
 {
 M2[branje] = 0;
 branje++;
 }
 branje = 0;
}
void UART_ST3_reset(void)
{
 while (branje < 100)
 {
 ST3[branje] = 0;
 branje++;
 }
 branje = 0;
}
```

```

int main(int argc, char **argv)
{
/*wiringPiSetupGpio();
pinMode(ledPin, OUTPUT);*/
 UART_M2_reset();
 UART_S1_reset();
 UART_ST3_reset();
 UART_Tx_reset();
 ST3_podatek = serialOpen("/dev/ttyAMA0", 9600);
 M2_podatek = serialOpen("/dev/ttyUSB0", 9600);
 S1_podatek = serialOpen("/dev/ttyUSB1", 9600);
 serialPuts(ST3_podatek, "ZORAN");
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10);
 delay(1000);
 // Pošiljanje TEST
 serialPuts(M2_podatek, "TEST");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 delay(50);
 while(1)
 {
 //////////////////////////////////////
 //////////////////////////////////////
 // BRANJE Vhoda
 stevilo = serialDataAvail(S1_podatek);
 while (stevilo > 0)
 {

 S1[branje] = serialGetchar(S1_podatek);
 branje++;
 stevilo = serialDataAvail(S1_podatek);
 // SP1++;
 }
 branje = 0;
 //////////////////////////////////////
 // Branje lzhoda (če bo slučanje potrebno)
 stevilo = serialDataAvail(M2_podatek);
 while (stevilo > 0)
 {

 M2[branje] = serialGetchar(M2_podatek);
 branje++;
 stevilo = serialDataAvail(M2_podatek);
 }
 branje = 0;
 //////////////////////////////////////
 // Branje telefona
 stevilo = serialDataAvail(ST3_podatek);
 while (stevilo > 0)
 {

 ST3[branje] = serialGetchar(ST3_podatek);
 branje++;
 stevilo = serialDataAvail(ST3_podatek);
 }
 branje = 0;
 //////////////////////////////////////
 // Zahteva za temperaturo
 if ((ST3[0] == 84 && ST3[1] == 77 && ST3[2] == 80 && ST3[3] == 63) || logika == 1)
 {
 if (enkrat == 0)
 {
 logika = 1;
 enkrat = 1;
 }
 }
 }
}

```

```

 UART_ST3_reset();
}
if(SP1 == 1)
{
 serialPuts(M2_podatek, "ABCTMP?");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 delay(350);
 SP1 = 0;
}
if(SP1 == 2)
{
 serialPuts(M2_podatek, "DEFTMP?");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 delay(350);
 SP1 = 0;
}
if(SP1 == 3)
{
 serialPuts(M2_podatek, "GHITMP?");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 delay(350);
 logika = 0;
 enkrat = 0;
}
}
if (S1[0] == 65 && S1[3] == 84 && S1[7] == 58 && S1[9] == 48)
{

 S1pretvorba = S1[10];
 S2pretvorba = S1[11];
 S3pretvorba = S1[12];
 S4pretvorba = S1[13];
 S1pretvorba = S1pretvorba - 48;
 S1pretvorba = S1pretvorba * 1000;
 S2pretvorba = S2pretvorba - 48;
 S2pretvorba = S2pretvorba * 100;
 S3pretvorba = S3pretvorba - 48;
 S3pretvorba = S3pretvorba * 10;
 S4pretvorba = S4pretvorba - 48;
 K1pretvorba = S1pretvorba + S2pretvorba + S3pretvorba + S4pretvorba;
 Kpretvorba = K1pretvorba + K2pretvorba + K3pretvorba;
 Kpretvorba = K1pretvorba + K2pretvorba + K3pretvorba;
 Kpretvorba = Kpretvorba / 3;
 Kpretvorba = Kpretvorba >> 4;
 SP1 = 2;

/* serialPuts(ST3_podatek, "Pret");
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10); */

}
if (S1[0] == 68 && S1[3] == 84 && S1[7] == 58 && S1[9] == 48)
{
 S1pretvorba = S1[10];
 S2pretvorba = S1[11];
 S3pretvorba = S1[12];
 S4pretvorba = S1[13];
 S1pretvorba = S1pretvorba - 48;
 S1pretvorba = S1pretvorba * 1000;
 S2pretvorba = S2pretvorba - 48;
 S2pretvorba = S2pretvorba * 100;
 S3pretvorba = S3pretvorba - 48;
 S3pretvorba = S3pretvorba * 10;

```

```

S4pretvorba = S4pretvorba - 48;
K2pretvorba = S1pretvorba + S2pretvorba + S3pretvorba + S4pretvorba;
Kpretvorba = K1pretvorba + K2pretvorba + K3pretvorba;
Kpretvorba = Kpretvorba / 3;
Kpretvorba = Kpretvorba >> 4;
SP1 = 3;
/* serialPuts(ST3_podatek, "Pret");
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10); */

}
if (S1[0] == 71 && S1[3] == 84 && S1[7] == 58 && S1[9] == 48)
{
 S1pretvorba = S1[10];
 S2pretvorba = S1[11];
 S3pretvorba = S1[12];
 S4pretvorba = S1[13];
 S1pretvorba = S1pretvorba - 48;
 S1pretvorba = S1pretvorba * 1000;
 S2pretvorba = S2pretvorba - 48;
 S2pretvorba = S2pretvorba * 100;
 S3pretvorba = S3pretvorba - 48;
 S3pretvorba = S3pretvorba * 10;
 S4pretvorba = S4pretvorba - 48;
 K3pretvorba = S1pretvorba + S2pretvorba + S3pretvorba + S4pretvorba;
 Kpretvorba = K1pretvorba + K2pretvorba + K3pretvorba;
 Kpretvorba = Kpretvorba / 3;
 Kpretvorba = Kpretvorba >> 4;
 // serialPuts(ST3_podatek, "Pret");
 // serialPutchar(ST3_podatek,13);
 //+ serialPutchar(ST3_podatek,10);
 printf("JAI G::\n");
 printf("%i °C \n<", Kpretvorba);
 SP1 = 1;

}

////////////////////////////////////
// Zahteva za svetlost
if ((ST3[0] == 83 && ST3[1] == 86 && ST3[2] == 69 && ST3[3] == 84 && ST3[4] == 63) || logika2 == 1)
{
 if (enkrat == 0)
 {
 logika2 = 1;
 enkrat = 1;
 UART_ST3_reset();
 }
 if (SP1 == 1)
 {
 serialPuts(M2_podatek, "ABCAN?");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 delay(350);
 SP1 = 0;
 }
 if (SP1 == 2)
 {
 serialPuts(M2_podatek, "DEFAN?");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 delay(350);
 SP1 = 0;
 }
 if (SP1 == 3)
 {
 serialPuts(M2_podatek, "GHIAN?");
 serialPutchar(M2_podatek,13);

```

```

serialPutchar(M2_podatek,10);
delay(350);
 logika2 = 0;
 enkrat = 0;
}
}
if (S1[0] == 65 && S1[3] == 83 && S1[7] == 58 && S1[9] == 48)
{

 S1pretvorba = S1[10];
 S2pretvorba = S1[11];
 S3pretvorba = S1[12];
 S4pretvorba = S1[13];
 S1pretvorba = S1pretvorba - 48;
 S1pretvorba = S1pretvorba * 1000;
 S2pretvorba = S2pretvorba - 48;
 S2pretvorba = S2pretvorba * 100;
 S3pretvorba = S3pretvorba - 48;
 S3pretvorba = S3pretvorba * 10;
 S4pretvorba = S4pretvorba - 48;
 K1pretvorba = S1pretvorba + S2pretvorba + S3pretvorba + S4pretvorba;
 SP1 = 2;

/* serialPuts(ST3_podatek, "Pret");
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10); */

}
if (S1[0] == 68 && S1[3] == 83 && S1[7] == 58 && S1[9] == 48)
{
 S1pretvorba = S1[10];
 S2pretvorba = S1[11];
 S3pretvorba = S1[12];
 S4pretvorba = S1[13];
 S1pretvorba = S1pretvorba - 48;
 S1pretvorba = S1pretvorba * 1000;
 S2pretvorba = S2pretvorba - 48;
 S2pretvorba = S2pretvorba * 100;
 S3pretvorba = S3pretvorba - 48;
 S3pretvorba = S3pretvorba * 10;
 S4pretvorba = S4pretvorba - 48;
 K2pretvorba = S1pretvorba + S2pretvorba + S3pretvorba + S4pretvorba;
 SP1 = 3;
/* serialPuts(ST3_podatek, "Pret");
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10); */

}
if (S1[0] == 71 && S1[3] == 83 && S1[7] == 58 && S1[9] == 48)
{
 S1pretvorba = S1[10];
 S2pretvorba = S1[11];
 S3pretvorba = S1[12];
 S4pretvorba = S1[13];
 S1pretvorba = S1pretvorba - 48;
 S1pretvorba = S1pretvorba * 1000;
 S2pretvorba = S2pretvorba - 48;
 S2pretvorba = S2pretvorba * 100;
 S3pretvorba = S3pretvorba - 48;
 S3pretvorba = S3pretvorba * 10;
 S4pretvorba = S4pretvorba - 48;
 K3pretvorba = S1pretvorba + S2pretvorba + S3pretvorba + S4pretvorba;
 Kpretvorba = K1pretvorba + K2pretvorba + K3pretvorba;
 Kpretvorba = Kpretvorba / 3;
// serialPuts(ST3_podatek, "Pret");
// serialPutchar(ST3_podatek,13);

```

```

 // serialPutchar(ST3_podatek,10);
 printf("JAIG\n");
 printf("%i Svetlost \n<", Kpretvorba);
 SP1 = 1;

}

////////////////////////////////////
// premik žaluzij
if (ST3[0] == 90 && ST3[1] == 65 && ST3[2] == 76 && ST3[3] == 85)
{
 serialPuts(M2_podatek, "JKLZALUA");
 if(ST3[4] == 'L')
 {
 serialPutchar(M2_podatek, 'L');
 }
 else if (ST3[4] == 'D')
 {
 serialPutchar(M2_podatek, 'D');
 }
 serialPutchar(M2_podatek, ST3[5]);
 serialPutchar(M2_podatek, ST3[6]);
 serialPutchar(M2_podatek, ST3[7]);

 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 UART_ST3_reset();
}
////////////////////////////////////
// vklop/izklop luči
if (ST3[0] == 76 && ST3[1] == 85 && ST3[2] == 67)
{
 /* serialPuts(ST3_podatek, "Luc sprememba stanja");
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10);*/
 if(ST3[3] == '1')
 {
 serialPuts(M2_podatek, "DEFLU0");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 }
 else if (ST3[3] == '0')
 {
 serialPuts(M2_podatek, "DEFLU1");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);
 }
}

 if(ST3[3] != 63)
 {
 UART_ST3_reset();
 }

}
////////////////////////////////////
// zahteva za luč

if (ST3[0] == 76 && ST3[1] == 85 && ST3[2] == 67 && ST3[3] == 63)
{
 // printf("adg\n");
 serialPuts(M2_podatek, "DEFLU?");
 serialPutchar(M2_podatek,13);
 serialPutchar(M2_podatek,10);

 UART_ST3_reset();
}

```

```

if (S1[0] != 0)
{
 serialPuts(ST3_podatek, S1);
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10);
 UART_S1_reset();

}

if (ST3[0] != 0)
{
 serialPuts(ST3_podatek, ST3);
 serialPutchar(ST3_podatek,13);
 serialPutchar(ST3_podatek,10);
 UART_ST3_reset();
}

delay(10);

}

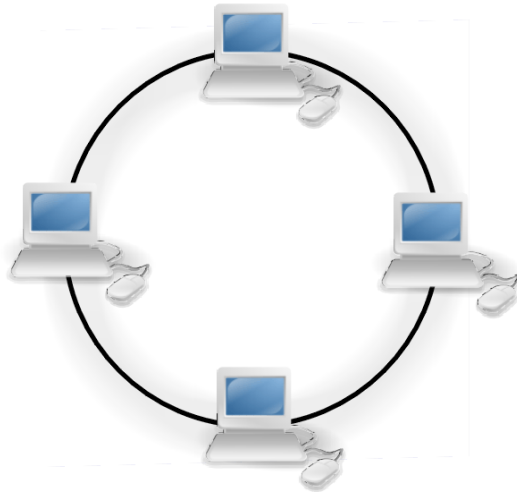
return 0;
}

```

## D. Računališke topologije

Poznamo šest računalniških topologij [3]:

- Krožna topologija – kjer je naprava povezana z dvema sosednjima. Smer toka podatkov je določen, zato vsaka naprava od ene sosednje prejema podatke in drugi pošilja.



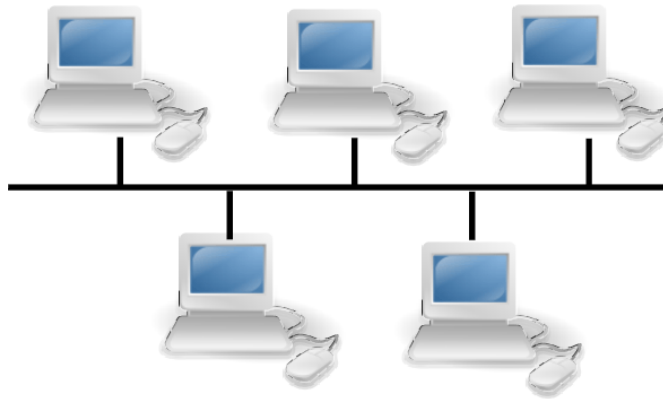
Slika 37: Krožna topologija

- Zvezdna topologija – kjer so vse naprave povezane direktno z eno centralo, ki regulira vso komunikacijo.



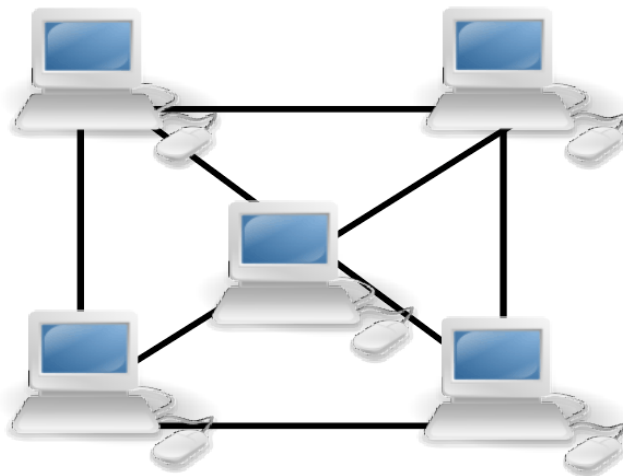
Slika 38: Zvezdna topologija

- Topologija vodila - vse naprave so povezane na istni žični vodnik.



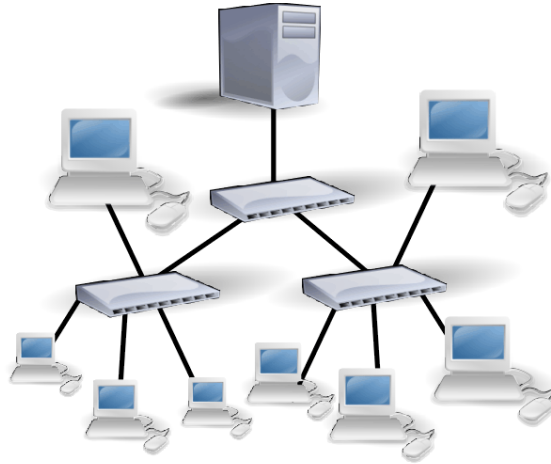
Slika 39: Topologija vodila

- Popolnoma povezana topologija - vsaka naprava je povezana z vsako.



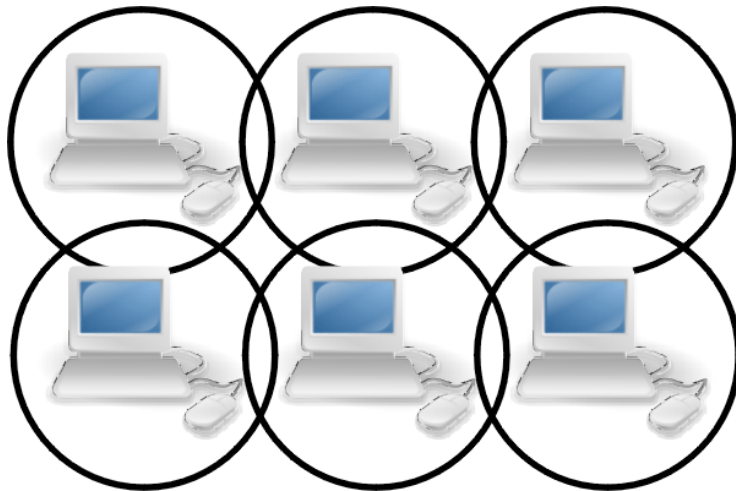
Slika 40: Topologija popolne povezave

- Hierhična topologija – naprave so razporejene hierhičnem sistemu.



Slika 41: Hiarhična topologija

- Mrežasta topologija- naprave so povezane poljubno tako, da ima vsaka naprava vsaj z eno povezana.



Slika 42: Mrežasta topologija